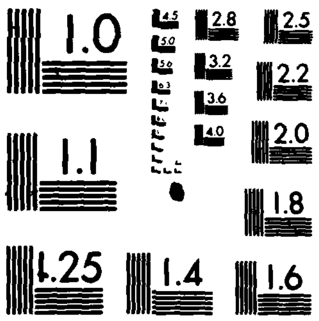
AD-A115 004   KAMAN AEROSPACE CORP BLOOMFIELD CT   F/G 9/2
DYNAMIC SYSTEM COUPLING (DYSCO) PROGRAM. VOLUME II. THEORETICAL--ETC(U)
APR 82   A BERMAN   DAAK51-79-C-0046
UNCLASSIFIED   R-1649-VOL-2   USAAVRADCOM-TR-81-D-42B   NL

1 of 1
A
115004

END
DATE
FILMED
7-82
DTIC


1.0
4.5
5.0
5.6
6.3
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

USAAVRADCOM-TR-81-D-42B

12

DYNAMIC SYSTEM COUPLING (DYSCO) PROGRAM
Volume II - Theoretical Manual

A. Berman
KAMAN AEROSPACE CORPORATION
Old Windsor Road
Bloomfield, Conn. 06002

April 1982

Final Report

Approved for public release;
distribution unlimited.

Prepared for
APPLIED TECHNOLOGY LABORATORY
U. S. ARMY RESEARCH AND TECHNOLOGY LABORATORIES (AVRADCOM)
Fort Eustis, Va. 23604

82 06 01 100

APPLIED TECHNOLOGY LABORATORY POSITION STATEMENT

This report documents the design, implementation, and use of a prototype computer program developed to demonstrate the feasibility of representing a rotorcraft system by dynamically coupling representations of individual system components and performing user specified solutions. Descriptions of hypothetical component, force, and solution algorithms and their use as required to treat a variety of typical rotorcraft problems not treatable with the prototype program and a technology complex based on the program design are provided. The design of Dynamic System Coupling program, DYSCO, should provide a sound basis for the development of comprehensive rotorcraft analysis systems.

The DYSCO program, though limited to consideration of articulated rotors having rigid blades and teetering rotors having rigid blades, can be used to obtain insight regarding the time history, mobility, and eigenanalysis characteristics of moderately complex rotorcraft configurations. Successful use of the DYSCO program, intended primarily for interactive use by a user familiar with the fundamental rotorcraft dynamic system, requires that the user be knowledgeable of the modeling and coupling techniques involved in the program.

Messrs. Paul Mirick, H. I. MacDonald, and Lawrence R. Sutton of the Aeromechanics Technical Area, Aeronautical Technology Division, provided technical direction for this effort.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER USAAVRADCOM TR 81-D-42B	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DYNAMIC SYSTEM COUPLING (DYSCO) PROGRAM Volume II - Theoretical Manual		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report
		6. PERFORMING ORG. REPORT NUMBER R-1649
7. AUTHOR(s) A. Berman		8. CONTRACT OR GRANT NUMBER(s) DAAK51-79-C-0046
9. PERFORMING ORGANIZATION NAME AND ADDRESS Kaman Aerospace Corporation Old Windsor Road Bloomfield, Connecticut 06002		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 612209 1L1 62209AH76 00 297 EK
11. CONTROLLING OFFICE NAME AND ADDRESS Applied Technology Laboratory, U.S. Army Research and Technology Laboratories (AVRADCOM) Fort Eustis, Virginia 23604		12. REPORT DATE April 1982
		13. NUMBER OF PAGES 82
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Volume II of a two-volume report

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Helicopter Analysis	Dynamics
Component Coupling	Aerodynamics
Computer Program	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Dynamic System Coupling (DYSCO) is a computer program which allows an interactive user to couple arbitrary components and force algorithms into a model of a helicopter or other dynamic system. The equations of the system may then be solved by a choice of analytical methods. The components available are rigid blade rotor, elastic fuselage, rotor control system, and other structures representable by general linear second-order differential equations. The force methods available are linear rotor loads, tabular rotor aerodynamics with optional induced velocity map, fuselage flat plate drag, and sinusoidal shaker.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Block 20. Abstract - continued.

The solution methods available are time history, linear constant coefficient eigenanalysis, and complex frequency response. The program has the capability of being expanded in its level of complexity by the addition of other technology modules.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

TABLE OF CONTENTS

TABLE OF CONTENTS - Continued

TABLE OF CONTENTS - Continued

2
F

LIST OF ILLUSTRATIONS

INTRODUCTION

DYSCO is a computer program for analyzing rotorcraft dynamic and aerodynamic phenomena based on coupling independent, arbitrary component representations into a valid representation of a complete system. Volume I of this report describes the general capabilities and use of the program and the specific capabilities of the technology modules.

This volume describes the mathematical basis, the specific coupling algorithms, the design, and the implementation of DYSCO and its potential capabilities.

The Contracting Officer's Representatives (Technical) have been Mr. Paul H. Mirick, Mr. Herman I. MacDonald, Jr., and Mr. Lawrence R. Sutton of the Applied Technology Laboratory. The participants in the contract effort were A. Berman, F. S. Wei, and N. Giansante of Kaman Aerospace Corporation, M. Anderson and P. Tyeryar of Control Data Corporation, and P. Baucom of P. B. Associates.

MATHEMATICAL BASIS OF DYSCO

The validity of DYSCO rests, in part, on the validity of the procedures for coupling the equations for the system components into the equations for the complete system.

ASSUMPTIONS

The general concept upon which DYSCO is based depends on the following assumptions.

1. The relevant physics of a rotorcraft may be modeled as a set of second-order differential equations in the time domain. These equations are of the form:

$$M\ddot{v} + C\dot{v} + Kv = F \tag{1}$$

where v is a vector of the degrees of freedom of the system. M,C, and K are coefficient matrices and F is a force vector where M,C,K, and F are arbitrary functions of $\dot{v}$,v, and time.

2. It is possible to formulate the equations of a system based on the equations of the components of the system. The equations of the components are of the same form as the equations of the system as in 1. above. "Formulation" includes the establishment of the logic which will allow the computation of the varying coefficients (elements of M,C, and K) and forces (F) of the system equations based on the computed coefficients and forces of each component at any point in time.

3. It is possible to compute the state vector ($\dot{v}$ and v) of each component based on the state vector of the system. The state vectors of the components may be used to compute the coefficients and forces of the equations of the components at each point in time. These coefficients and forces may then be used to compute the coefficients and forces of the system equations as in 2. above.

COMPONENT EQUATIONS

Any portion of the system which can be represented by a set of second-order differential equations, subject to the conditions stated below, may be considered to be a "component". The equations of component i may be written

$$M_i\ddot{v}_i + C_i\dot{v}_i + K_iv_i = F_i \tag{2}$$

where v_i is a vector containing the degrees of freedom of the component.

The coefficient matrices, M_i, C_i, and K_i, and the generalized force vector, F_i, may be arbitrary functions of time, $\dot{v}_i$, and v_i. The elements of F_i may also be functions of $\dot{v}$ and v of other components. "Arbitrary function" as used above means that it is possible to write a program to compute the numerical value of the function at any point in time given numerical values of the appropriate variables. This functionality includes linear, nonlinear analytical, periodic, and tabular functions.

Consistent with the definition of F_i, Eq. (2) may also be written as

$$M_i \ddot{v}_i = F_i \tag{3}$$

where F_i (Eq. (3)) $\equiv F_i$ (Eq. 2)) $- C_i \dot{v}_i - K_i v_i$. For reasons of computational efficiency and to allow for quasistatic stability or quasilinear eigenanalysis, the more general form of Eq. (2) shall be used in this report. In any particular application, however, C_i and K_i may be null matrices or any portions of them may be included in F_i.

The individual degrees of freedom must represent independent (generalized) coordinates. The degrees of freedom may be actual physical displacements in coordinate systems which are fixed or moving at a uniform velocity or rotating at a constant velocity. They may also represent modal displacements or any other generalized coordinate. v_i may contain any combination of the allowable types of degrees of freedom.

The coupling treated in DYSCO involves equating physical displacements of two or more components. These physical displacements may or may not be generalized coordinates of the components, however, they must be expressible as a linear combination of degrees of freedom of each of the components to be coupled. Thus the vector, v_i, must contain appropriate degrees of freedom to allow desired coupling to other components.

SYSTEM EQUATIONS

The equations of a system consisting of one or more components, each represented by a set of equations of the form of Eq. (2), may be represented in the same form:

$$M\ddot{v} + C\dot{v} + Kv = F \tag{4}$$

The system coefficient matrices and generalized force vector are formed from the component matrices and vectors as shown below. The degree of freedom vector of the system, v, must contain only independent coordinates. This vector may be formed by assembling the degrees of freedom of all the components and eliminating duplications when specific degrees of freedom are coupled or eliminating a degree of freedom whenever linear combinations of degrees of freedom are coupled.

COORDINATE TRANSFORMATIONS

Each degree of freedom of a component is equal to a degree of freedom of the system or is a linear combination of degrees of freedom of the system. This relationship may be expressed as a matrix equation:

$$v_i = T_i\, v \tag{5a}$$

where T_i is an n_i x n matrix, and where n_i and n are the numbers of degrees of freedom of the component and the system. Since the coupling treated in DYSCO physically joins components, T_i may not be a function of time and therefore

$$\dot{v}_i = T_i\, \dot{v} \tag{5b}$$

$$\ddot{v}_i = T_i\, \ddot{v} \tag{5c}$$

When the system contains degrees of freedom in fixed and moving coordinate systems, any necessary transformations are to be modeled within the component representations. This will result in time dependent coefficients which is consistent with the component equations as previously defined.

DERIVATION BASED ON PHYSICAL CONSTRAINTS

Using the transformation relationships, Eq.(5), the equation for a component (Eq.(2)) may be written in terms of the system degrees of freedom:

$$M_iT_i\ddot{v} + C_iT_i\dot{v} + K_iT_i\, v = F_i \tag{6}$$

This matrix equation, however, is not a proper dynamic representation since the coefficient matrices are rectangular and F_i is not the generalized force vector corresponding to the degrees of freedom, v. The generalized force vector is defined as the vector that yields work done by the external forces when the product $v_i{}^TF_i$ is formed where $v_i{}^T$ is the transpose of v_i.

The work done cannot depend on the coordinate system; that is, the work due to the component degrees of freedom must be equal to the work due to the system degrees of freedom. Then since $v_i{}^TF_i = v^TT_i{}^TF_i = v^T(T_i{}^TF_i)$, the right-hand side of the transformed matrix equation must be $T_i{}^TF_i$.

This is achieved by premultiplying Eq. (6) by T_i^T, resulting in

$$T_i^T M_i T_i \ddot{v} + T_i^T C_i T_i \dot{v} + T_i^T K_i T_i v = T_i^T F_i \tag{7}$$

When this transformation is performed on each component set of equations, they are consistent and may be added to form the equations of the coupled system:

$$M\ddot{v} + C\dot{v} + Kv = F \tag{8}$$

where

$$\begin{aligned} M &= \sum_{i=1}^{n} T_i^T M_i T_i \\ C &= \sum_{i=1}^{n} T_i^T C_i T_i \\ K &= \sum_{i=1}^{n} T_i^T K_i T_i \\ F &= \sum_{i=1}^{n} T_i^T F_i \end{aligned} \tag{9}$$

These equations are identical with other representations (e.g., Hurty[1]), except that they are not limited to linear equations. It should be noted that, when the coefficients and forces are functions of time or nonlinear, Eq. (9) may be evaluated numerically only at discrete values of time or when the component state vectors ($\dot{v}_i$ and v_i) have been evaluated if the coefficients or forces are nonlinear.

1. Hurty, W. C., "Dynamic Analysis of Structural Systems by Component Mode Synthesis, "Jet Propulsion Lab., Pasadena, CA, Technical Report 32-530, Jan. 15, 1964.

TRANSFORMATION METHODS

In the previous section the transformation of component coefficient matrices, force vectors, and degree of freedom vectors was defined by use of a set of matrices, T_i, each associated with a particular component.

These transformation matrices are used for two purposes: (1) to transform the component coefficients and forces into system coefficients and forces (Eq.(9)); and (2) to retrieve the component state variables from the system state variables (Eq. (5)). The transformation matrices are inherently very sparse, and storing and operating with them as ordinary matrices would be extremely wasteful of storage and computationally inefficient. Thus, special algorithms were developed for the implementation of DYSCO.

The coordinate transformations used in DYSCO are of two types: (1) a component degree of freedom is equal to a system degree of freedom ("explicit"); (2) a component degree of freedom is equal to a linear combination of system degrees of freedom ("implicit").

Consider, first, the simpler case of explicit coupling. This is in reality a special case of implicit coupling where a single nonzero coefficient exists in the relevant row of the transformation matrix and is equal to unity.

EXPLICIT COUPLING

When structures are coupled by joining physical degrees of freedom of components and where the displacements have been aligned so that the degrees of freedom are defined in the same direction, then this degree of freedom of each of the joined components will be equal to a degree of freedom of the system. Also included in this category of "explicit" coupling are component degrees of freedom which are not coupled (sometimes called "interior" degrees of freedom) but retained in the system equations of motion.

In DYSCO an integer two-dimensional array, TRAN, is formed which has a separate column for each component. Each column contains an element for each component degree of freedom. The value of an element of TRAN is the index of the corresponding system degree of freedom.

As an illustration, consider the schematic of a simple system in Figure 1. Note that, in this figure, the coupling is indicated by like names of the respective degrees of freedom of the components.

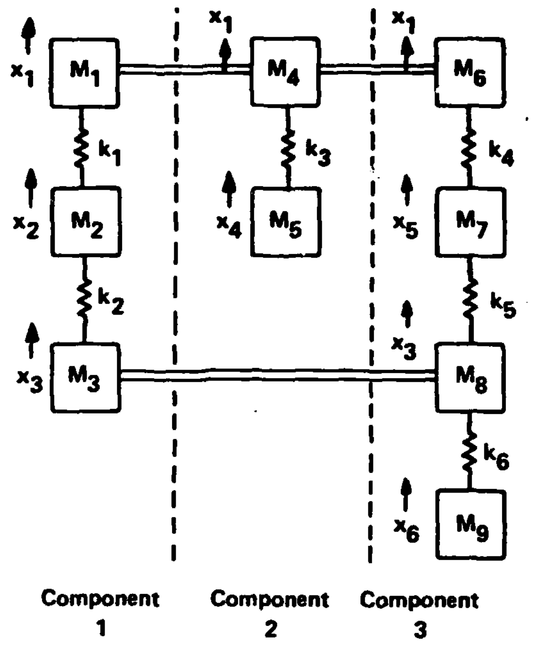


Figure 1. Simple Three Component Coupled System.

The degree of freedom vectors for this system are

$$v_1 = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad v_2 = \begin{bmatrix} x_1 \\ x_4 \end{bmatrix} \quad v_3 = \begin{bmatrix} x_1 \\ x_5 \\ x_3 \\ x_6 \end{bmatrix} \quad v = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} \tag{10}$$

The transformation matrices are (Eq.(5a))

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{11}$$

$$T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{12}$$

$$T_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

The transformation array used in DYSCO would be

$$\mathrm{TRAN} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 5 \\ 3 & 0 & 3 \\ 0 & 0 & 6 \end{bmatrix} \qquad (14)$$

The use of TRAN to retrieve component degrees of freedom from the system vector is illustrated by

$$v_i(j) \quad = \quad v(\mathrm{TRAN}(j,i)) \qquad (15)$$

That is, the jth degree of freedom of the ith component is equal to the TRAN(j,i)th degree of freedom of the system.

The array TRAN (Eq. (14)) uses 12 words to include all the information in the matrices of Eq. (11), (12), and (13) which contain 54 words. For a case of moderate complexity assume $n_1 = 20$, $n_2 = 10$, $n_3 = 10$, $n_4 = 30$, and $n = 40$. The transformation matrices would contain 70x40 or 2800 pieces of data and TRAN would contain 30x4 or 120 words. (If the columns of TRAN were "packed" only 70 words would be required.)

In addition to retrieving component degrees of freedom from system degrees of freedom, TRAN may also be used in a very simple fashion to transform the component coefficients and forces to system coefficients and forces as shown below.

If C is a component coefficient matrix of component I, S is the corresponding system matrix equal to $T_I^T C_I T_I$, and NI is the number of degrees of freedom of the component, then the transformation may be written as a FORTRAN algorithm, as follows:

```
          DO 2Ø  J  =  1, NI
          DO 1Ø  K  =  1, NI
          S(TRAN(J,I), TRAN(K,I) ) = C(J,K)
      1Ø  CONTINUE
      2Ø  CONTINUE
```

S is summed over all the system components to obtain the system coefficient matrix. This process is performed for each of the M, C, and K matrices. The corresponding transformation for the forcing function is

```
          DO 1Ø  J  =  1, NI
          SF(TRAN(J,I)) = F(J)
      1Ø  CONTINUE
```

IMPLICIT COUPLING

In DYSCO the term "implicit" coupling is used to refer to the case where a physical degree of freedom of one component is equal to a linear combination of degrees of freedom of another component. This is equivalent to the common term "multipoint constraint" used in finite element analysis. In DYSCO both explicit and implicit couplings are allowed. The array TRAN is used as described above, except that when an implicit coupling exists a negative integer in TRAN directs the program to two packed arrays: SCO, which contains the appropriate multipliers (real); and SCODF, which contains the corresponding system degree of freedom indices (integer). The absolute value of the negative integer in TRAN is the starting index of SCO and SCODF and the series continues until a negative value in SCODF signals the end.

As a simple illustration, consider the system of Figure 2 where a lumped mass component is attached to a modal beam representation.

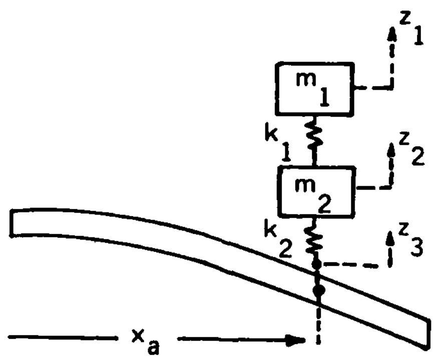


Figure 2. Example of Implicit Coupling.

For this system, the component and system degree of freedom vectors are

$$v_1 = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \qquad v_2 = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \qquad v = \begin{bmatrix} z_1 \\ z_2 \\ q_1 \\ q_2 \end{bmatrix} \tag{16}$$

where the beam vertical deflection is represented by two modes, ϕ_1 and ϕ_2, with corresponding generalized coordinates q_1 and q_2.

The implicit relationship is

$$z_3 = \phi_1(x_a)\, q_1 + \phi_2(x_a)\, q_2 \tag{17}$$

where $\phi_1(x_a)$ and $\phi_2(x_a)$ are the numerical values of the normalized modal displacements at the attachment point, x_a. In the coupled system, the redundant coordinate z_3 is eliminated.

The transformation matrices for this example are

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \phi_1(x_a) & \phi_2(x_a) \end{bmatrix} \tag{18}$$

$$T_2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{19}$$

In the DYSCO program, the transformation arrays would be

$$\text{TRAN} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \\ -1 & 0 \end{bmatrix} \tag{20}$$

$$\text{SCO} = \begin{bmatrix} \phi_1(x_a) \\ \phi_2(x_a) \end{bmatrix} \quad \text{SCODF} = \begin{bmatrix} 3 \\ -4 \end{bmatrix} \tag{21}$$

The positive integers in TRAN are treated as in the previous section. When the transformation for z_3 is sought, TRAN(3,1) (3rd degree of freedom of 1st component) is -1. This directs the computer algorithm to element 1 of SCO and SCODF and the relationship

$$\begin{aligned} v_1(3) &= \text{SCO}(1)*v(|\text{SCODF}(1)|) \\ &\quad +\text{SCO}(2)*v(|\text{SCODF}(2)|) \\ &= \phi_1(x_a)*v(3)+\phi_2(x_a)*v(4) \end{aligned} \tag{22}$$

where $\phi_1(x_a)$ and $\phi_2(x_a)$ are numerical values. The series ends when SCODF(2) = -4 is reached.

Figure 3 illustrates a more complex coupled system consisting of six components. Components 2 and 5 are represented by the use of modal degrees of freedom and the other components have physical degrees of freedom. Explicit coupling is indicated by solid lines and common names, implicit coupling is indicated by dashed lines. The linear relationships are defined in the figure.

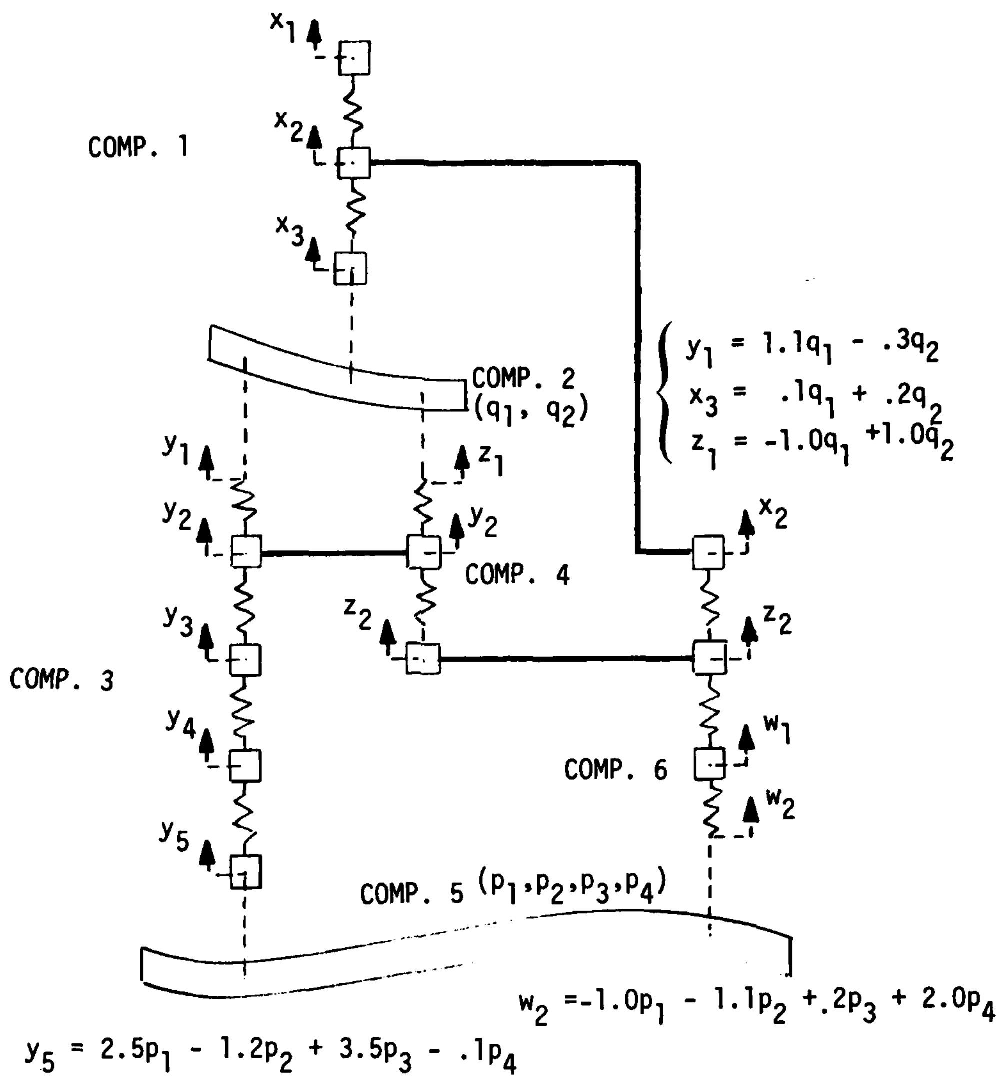


Figure 3. More Complex System.

The component and system degrees of freedom for this example are shown below.

Variable	v_1	v_2	v_3	v_4	v_5	v_6	v
1	x_1	q_1	y_1	z_1	p_1	x_2	x_1
2	x_2	q_2	y_2	y_2	p_2	z_2	x_2
3	x_3		y_3	z_2	p_3	w_1	q_1
4			y_4		p_4	w_2	q_2
5			y_5				y_2
6							y_3
7							y_4
3							z_2
9							p_1
10							p_2
11							p_3
12							p_4
13							w_1

The elements of the corresponding TRAN, SCO, and SCODF arrays are

TRAN

Variable	1	2	3	4	5	6
1	1	3	-3	-9	9	2
2	2	4	5	5	10	8
3	-1	0	6	8	11	13
4	0	0	7	0	12	-11
5	0	0	-5	0	0	0

Index	SCO	SCODF
1	.1	3
2	.2	-4
3	1.1	3
4	- .3	-4
5	2.5	9
6	-1.2	10
7	3.5	11
8	- .1	-12
9	-1.0	3
10	1.0	-4
11	-1.0	9
12	-1.1	10
13	.2	11
14	2.0	-12

If the data were stored in T matrices, 21x13 or 273 words would be required. TRAN, SCO, and SCODF combined use 58 storage words.

As an illustration of the use of these arrays for retrieving a component degree of freedom variable from the system degree of freedom variables, consider the following cases:

1. *Find y_2. y_2 is variable 2 of component 4.* TRAN(2,4) = 5. Since this value is positive, y_2 = variable 5 of the system degrees of freedom (i.e., $v_2(2) = v(5)$).

2. Find w_2. w_2 is variable 4 of component 6. TRAN(4,6) = -11. Since this is negative, start at element 11 of SCO and SCODF and form the linear combination until SCODF becomes negative. That is, $v_6(4) = 1.0 * v(9) - 1.1 * v(10) + .2 * v(11) + 2.0 * v(12)$.

The TRAN, SCO, and SCODF arrays are also used to transform the component coefficients and forces to the system coefficients and forces (Eq. (9)). This process is only slightly more complicated than the process described above and is performed in subroutines XXCCM and XXCCF of DYSCO.

The TRAN, SCO, and SCODF arrays are formed automatically by DYSCO based on information supplied during the "definition phase" of each component. This procedure is discussed in a later section.

OVERALL DESIGN OF DYSCO

The design of DYSCO is based on the requirements for a general purpose rotorcraft analytical simulator. The specific design is greatly influenced by the theoretical basis of DYSCO and by the specific capabilities of the algorithms which were developed. The major abilities of the DYSCO algorithms which have influenced the DYSCO design are:

1. Complete separation of individual component representations from other component representations.
2. Automatic formulation of transformation arrays and system degrees of freedom.
3. Retrieval of component state variables from system state variables at each time step (during time domain solution).
4. Computation of component coefficients and forces at each time step.
5. Transformation of the component coefficients and forces into the system coefficients and forces at each time step.

The above capabilities have been achieved in DYSCO by the separation of each component representation (component technology module) into several program modules (component technical modules) which appear in different locations in the DYSCO design. This technical module concept is vital to the design implemented and will be briefly discussed here prior to the description of the architecture of DYSCO.

COMPONENT TECHNICAL MODULES

A DYSCO "component" is any part of a rotorcraft analytical model which is represented by a set of component technical modules. A library of sets of component technical modules may contain several representations of a physical component, each set having a different level of complexity or a different analytical formulation.

In general, the user of the DYSCO program may select any combination of the available components to form a "model".

In DYSCO each component is represented by four technical modules which appear in different phases of the program:

1. Input Module. This technical module accepts all the input required by the other modules associated with the component.

2. Definition Module. This technical module defines the degrees of freedom and implicit relationships related to the particular use of the component representation.

3. Coefficient Module. This technical module computes the constant coefficients and forces of the component equations.

4. Active Module. This technical module computes the time varying coefficients and forces of the component equations based on the component state vector at any point in time. The active module accesses an optional "force module" specified by the user for the computation of applied forces.

The first three modules are used only once in the formation and solution of a set of system equations. The active module is used repeatedly during a time history solution.

These modules are executed independently but share data. The input module forms data sets which are placed on a run data file (RDF). The other three modules share pertinent data through a labeled COMMON area which is specific to the particular component representation. This is discussed in a later section.

FIRST LEVEL DYSCO DESIGN

The basic design of DYSCO is shown in Figure 4 as consisting of four major execution modules. The functions of these modules are briefly described below.

1.0 Preprocess Data. This execution module carries out operations specified by the users reference to specific preprocessing modules which are independent of the coupled configuration. Examples are: blade modal computation, reductions in coordinates of fuselage finite element model, and computation of rigid wake inflow map. (A "preprocess data" execution module has not been implemented in DYSCO.)

2.0 Create New Model. This execution module accepts control inputs specifying the specific component and force technology modules to be included in the rotorcraft system model and accepts input data defining component degrees of freedom and necessary physical parameters and accepts input specifying the solution to be executed. Data sets are created and referenced which contain the data required for the component technical modules used in 3.0 below. In addition data defining the sequence of execution of component technical modules are generated.

3.0 Execute Model. This execution module forms the component and system equations and carries out the specified solution.

4.0 Postprocess Data. This execution module performs operations on data obtained during the solution of the system equations. Examples are: harmonic analysis of blade bending moments, noise analysis based on rotor pressure distribution, and the calculation of internal loads of a component. (A "postprocess data" module has not been implemented in DYSCO.)

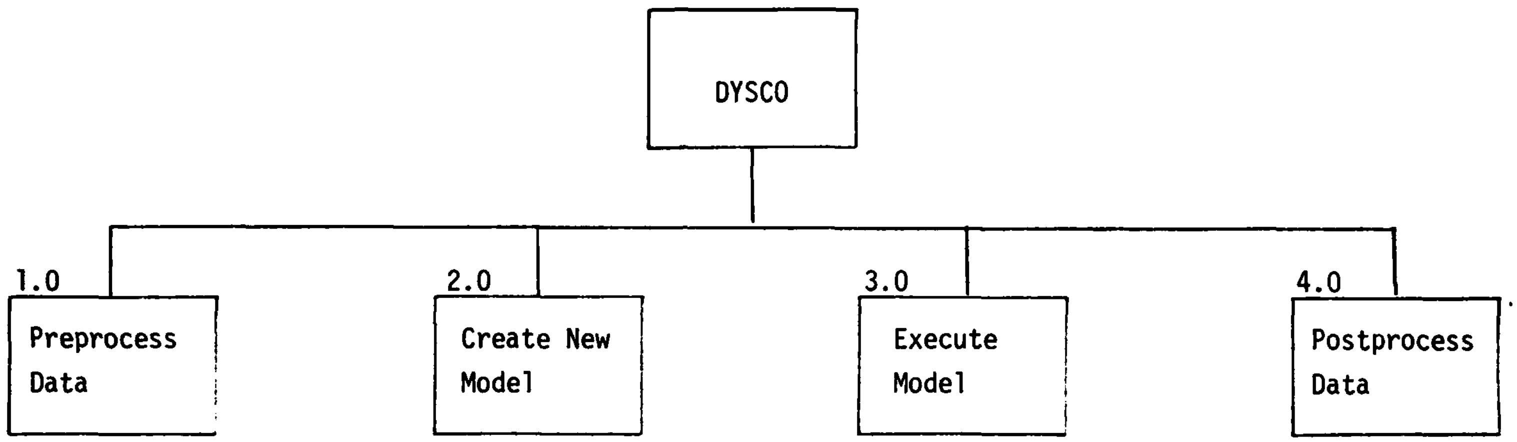


Figure 4. First Level DYSCO Design.

The two implemented execution modules are discussed in greater detail below.

Create New Model (2.0)

This execution module gathers all the information which completely defines an analytical model. Its major functions are as shown in Figure 5 and described below.

2.1 Input Model Name. Accepts user input of a unique model name which identifies a data set on the run data file for future reference.

2.2 Input Component, Force Information. Accepts user input of a component technology module name, force technology module name, and associated data set names (new or old) (2.2.1). The appropriate component and force input technical modules (2.2.2.1, 2.2.2.2) are accessed if the data set names are new and input is accepted to form the corresponding data sets of the run data file associated with the model name (2.1). This input process is repeated until all components of the model are specified.

2.3 Form Sequence Table Data. These data consist of a list of the ordinals identifying the components, the corresponding user supplied rotor or structure numbers, the integers indicating the number of the reference to the same component, and the logical unit and record numbers associated with the input data for the component definition and coefficient modules and a similar set of data for the named force modules. These data are the program control information required to execute the model in the solution phase. The sequence table data are discussed in more detail in a later section.

2.4 Input Solution Name. Accepts user input of a solution technology module name.

Execute Model (3.0)

This execution module forms the system equations and performs the specified solution. The hierarchy of this phase is shown in Figure 6 and described below.

3.1 Form Model. In "create new model", above, all the data required were assembled. In this module these data are used to define the component and system degrees of freedom and form the required transformation arrays (3.1.1), to compute the component and system constant coefficients and forces (3.1.2), and to compute any constant parameters required by the corresponding force module (3.1.3).

3.2 Execute Solution. In this module, the specified solution input data are accepted (3.2.1), the solution is carried out (3.2.2), and the results are made available (3.2.3).

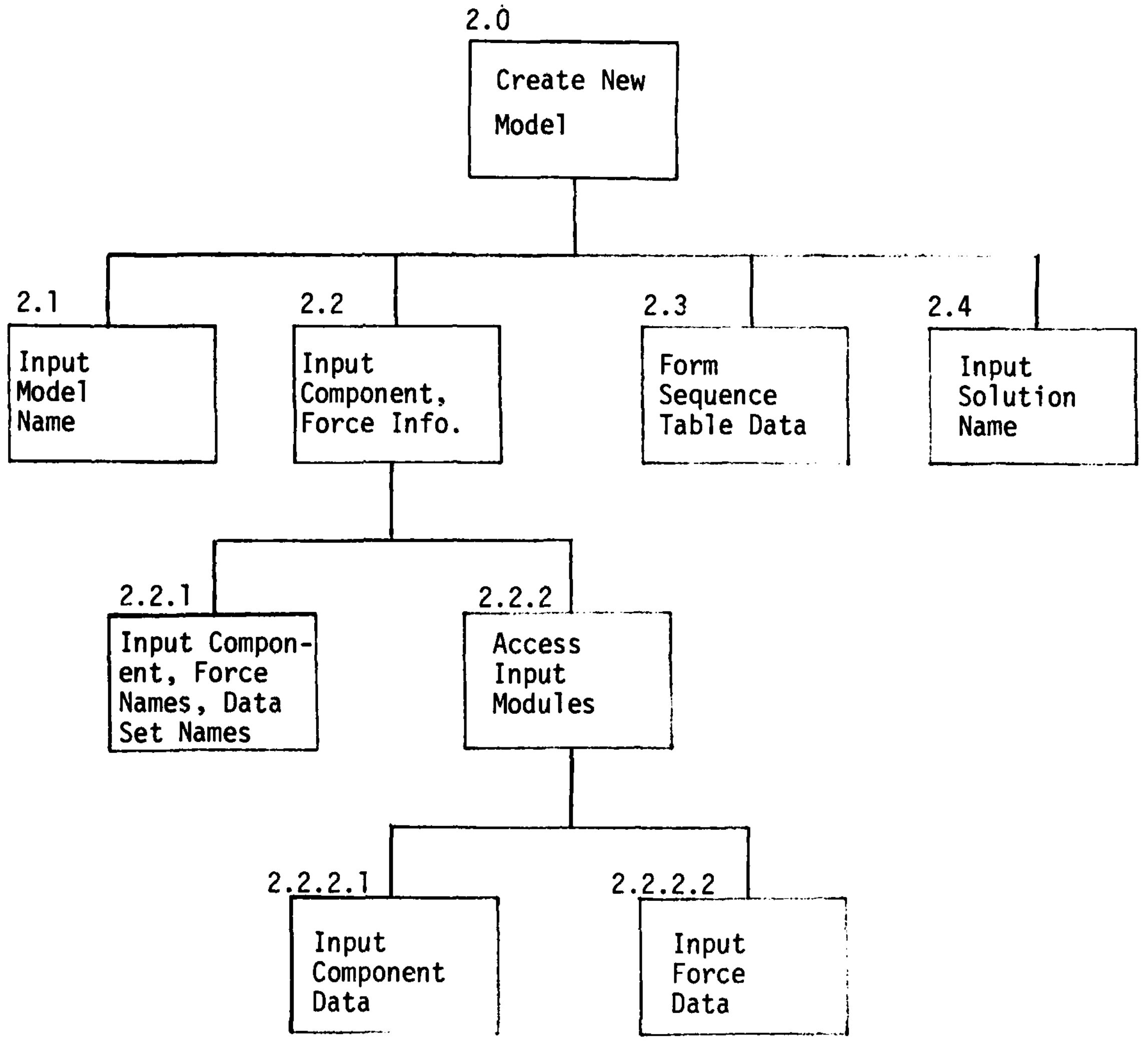


Figure 5. "Create New Model" Hierarchy.

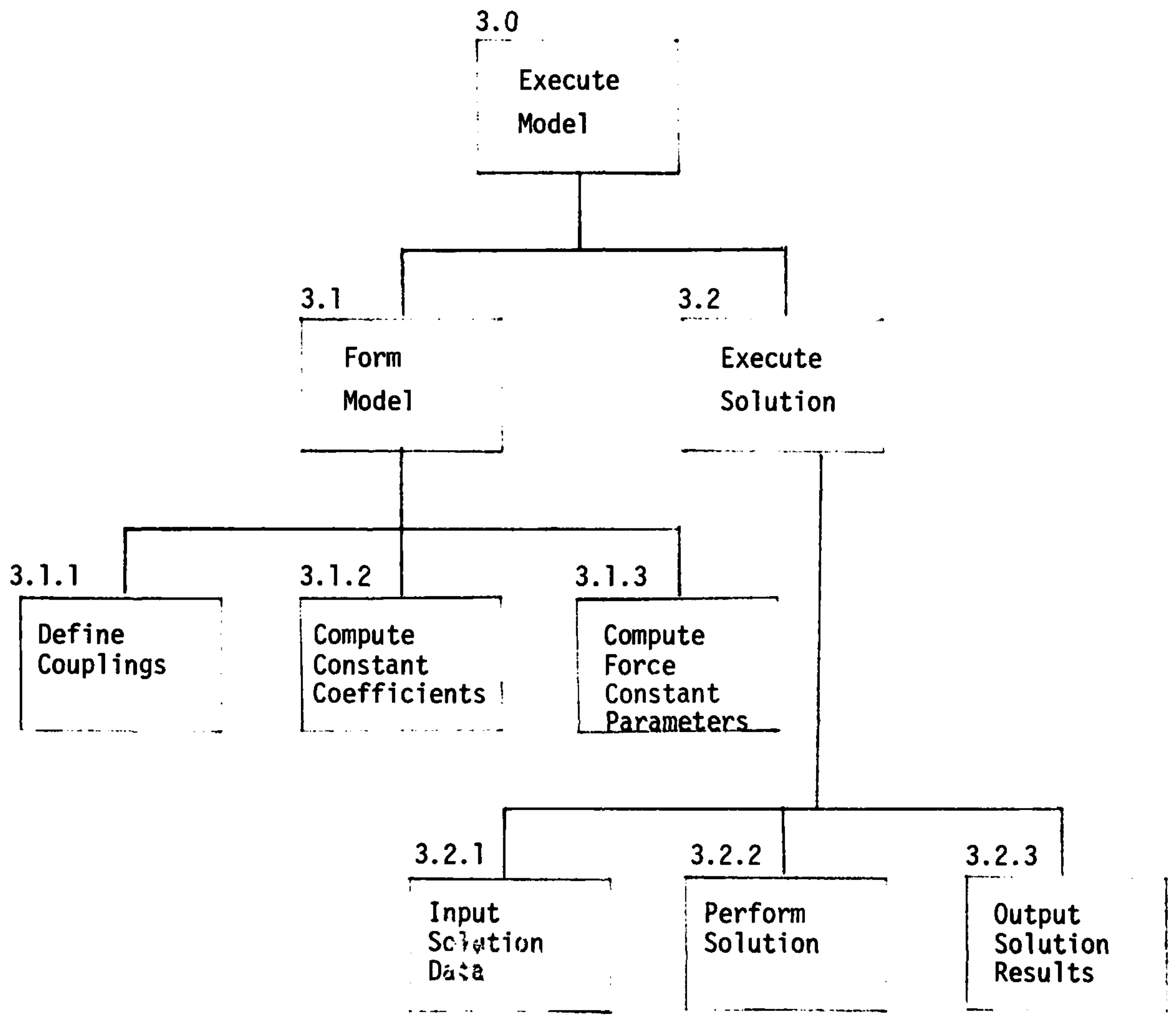


Figure 6. "Execute Model" Hierarchy.

The "define couplings" module (3.1.1) is shown in Figure 7 and the "compute constant coefficients" module (3.1.2) is shown in Figure 8. Note that 3.1.1.1 and 3.1.2.1 are specific component "definition" and "coefficient" technical modules. 3.1.2.1 computes the constant coefficients of M_i,C_i,K_i, and F_i of Equation (2). 3.1.2.2 transforms M_i, C_i, and K_i and 3.1.2.3 transforms F_i to the system equations.

In the "perform solution" module (3.2.2) the component "active" technical modules are accessed. The available solution methods are discussed in a later section.

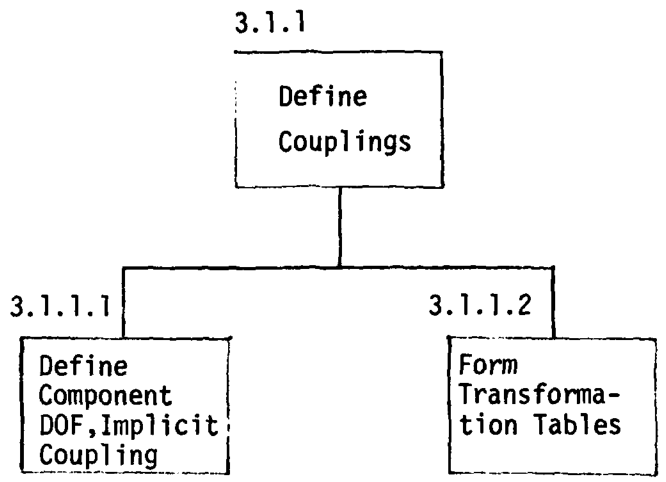

3.1.1
Define Couplings
3.1.1.1
Define Component DOF,Implicit Coupling
3.1.1.2
Form Transformation Tables

Figure 7. "Define Couplings" Hierachy.

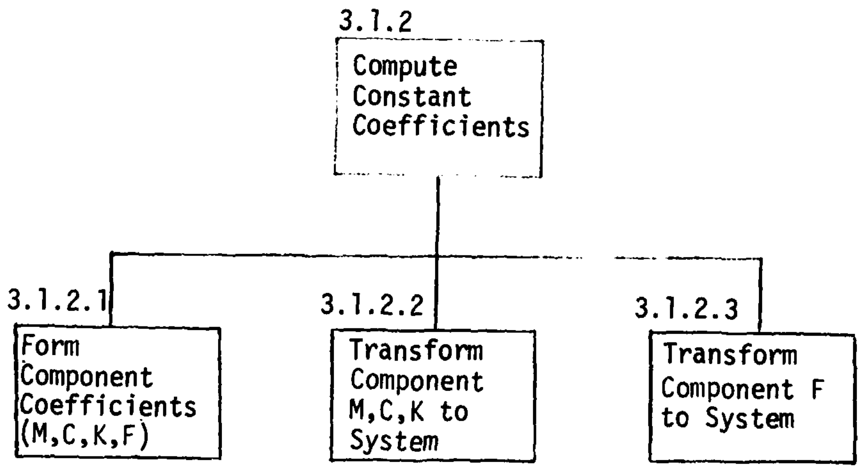

3.1.2
Compute Constant Coefficients
3.1.2.1
Form Component Coefficients (M,C,K,F)
3.1.2.2
Transform Component M,C,K to System
3.1.2.3
Transform Component F to System

Figure 8. "Compute Constant Coefficients" Hierarchy.

IMPLEMENTATION OF DYSCO

Several features of the implementation of DYSCO are worthy of a more detailed explanation: standardization of the names of the technology and technical modules and the names of the component degrees of freedom; definition of the program control variables which control the execution of the component technical modules; further details of the definition, coefficient, and solution phases; and sharing of data among modules. These are treated in the following sections and examples are given for purposes of illustration. In addition, technical module specifications and DYSCO hierarchy charts are presented.

MODULE NAMES

The technology modules; each containing the representation for either a component, applied forces, or a solution procedure; have a four character name. The first character of a technology module name may be C, F, or S corresponding to a component, force, or solution technology module, respectively. The second and third character are general descriptors and the fourth character is an integer indicating the level of complexity. The technology modules implemented in DYSCO are:

CSF1	-	Structure, finite element
CFM2	-	Fuselage, modal
CRR2	-	Rotor, rigid blade
CCE1	-	Control system, elastic
FRAØ	-	Rotor aerodynamics, linear
FFAØ	-	Fuselage aerodynamics, flat plate
FSS1	-	Structure, shaker
FRA2	-	Rotor aerodynamics, tabular
STH3	-	Time history
SFD1	-	Frequency domain
SEA3	-	Eigenanalysis

These names are used to refer to the specific components, forces, and solutions in a model formulation. Detailed descriptions of these technology modules are provided in Volume I of this report.

Each technology module consists of three or four technical modules.

The component technical modules (implemented as subroutines in DYSCO) have names of the following form:

C---I
C---D
C---C
C---A

where the first four characters correspond to the associated component technology module name. The fifth character is an I,D,C, or A indicating a component input, definition, coefficient, or active technical module, respectively.

Similarly, the force technical modules (implemented as subroutines) are named

```
F---I
F---C
F---A
```

and the solution technical modules (implemented as subroutines) are named

```
S---I
S---A
S---O
```

where, in general, the suffix I indicates input, C indicates coefficient computation, A indicates "active" or "analysis", and O indicates output.

DEGREE OF FREEDOM NAMES

In DYSCO the coupling is performed by interfacing degrees of freedom having
5 identical variable names. The standard names are of the form of four
F alpha characters plus four integer characters. In most component technology modules the degree of freedom names are automatically formed.

Each rotor (and rotor control system) component has associated with it a unique "rotor number" supplied by the user, and each structure (CSF1 or CFM2) has a unique "structure number" supplied by the user.

The degree of freedom names are defined in Volume I of this report and are listed here for purposes of illustration.

The degree of freedom (DOF) names associated with each component are (where r = rotor number, b = blade number, s = structure number):

<u>CRR2</u> up to 6 hub DOF in the fixed system:

XHUBr000	-	aft displacement
YHUBr000	-	starboard displacement
ZHUBr000	-	vertical displacement
ALFXr000	-	roll angle
ALFYr000	-	pitch angle
ALFZr000	-	yaw angle

and up to 3 DOF per blade in the rotating system:

BETArb00	-	flap angle
ZETArb00	-	lag angle
THETrb00	-	pitch angle

Provisions are included for coupling to a control system through the implicit relationship

RODRrb00 = PHL * THETrb00

where RODRrb00 are the control rod DOF of component CCE1 and PHL is the pitch horn length.

CCE1 3 swashplate DOF in the fixed system:

ZSPTr000	-	collective (axial) displacement
ASPXr000	-	roll angle
ASPYr000	-	pitch angle

and 1 DOF per control rod in the rotating system:

RODRrb00 - displacement at upper end of rod

CSF1 All DOFs are arbitrary and supplied by user.

CFM2 up to 6 rigid body modes:

XCG s000	-	aft translation
YCG s000	-	starboard translation
ZCG s000	-	vertical translation
ROLLs000	-	roll angle
PTCHs000	-	pitch angle
YAW s000	-	yaw angle

and up to 6 elastic coupled modes:

QFUSsm00 - m = mode number

Provisions are included for general rotor hub implicit attachments and other arbitrarily named DOFs. Implicit relationships are automatically formulated based on modal and rigid body displacements.

PROGRAM CONTROL VARIABLES

Some of the more important variables and terms associated with the control of the DYSCO program are defined below.

For the purposes of illustration, assume that a DYSCO user has specified the following components and force modules in the sequence shown.

I	COMPONENT	FORCE	ROTOR OR STRUCTURE NUMBER
1	CRR2	FRA2	1
2	CCE1	none	1
3	CSF1	FFAØ	1
4	CSF1	none	2
5	CRR2	FRA2	2
6	CSF1	FSS1	3
7	CCE1	none	2

The variables related to program control are as follows:

NC — Number of components (In the example, NC=7.)

CORDA(I) — Integer array of the ordinals associated with the components, where

CSF1	=	1
CFM2	=	2
CRR2	=	4
CCE1	=	5

(In the example, CORDA = 4,5,1,1,4,1,5.)

CREFA(I) — Integer array of the number of the reference to the Ith component

(In the example, CREFA = 1,1,1,2,2,3,2. CREFA(4) =2 for example, denotes the second reference to CSF1.)

FORDA(I) — Integer array of the ordinals associated with the force modules where

FRAØ	=	1
FFAØ	=	2
FSS1	=	3
FRA2	=	4
none	=	0

(In the example, FORDA = 4,0,2,0,4,3,0.)

FREFA(I) — Integer array of the number of the reference to the Ith force module
(In the example, FREFA = 1,0,1,0,2,1,0.)

NUMA(I) — Integer array of the rotor or structure numbers supplied for the components
(In the example, NUMA = 1,1,1,2,2,3,2.)

Summarizing the above example:

I	CORDA	CREFA	FORDA	FREFA	NUMA
1	4	1	4	1	1
2	5	1	0	0	1
3	1	1	2	1	1
4	1	2	0	0	2
5	4	2	4	2	2
6	1	3	3	1	3
7	5	2	0	0	2

This information comprises, in part, what might be called a "sequence control table". Other necessary information stored in integer arrays defines the location of data:

CLUN(I)
CREC(I,1)
CREC(I,2)
	Logical unit number and record numbers for the files containing the "definition" and "coefficient" data for component I

FLUN(I,1)
FREC(I,1)
FLUN(I,2)
FREC(I,2)
	Logical unit and record numbers for the files containing basic force data and auxillary force data (e.g., aerodynamic tables) for component I

DEGREE OF FREEDOM VARIABLES

FORTRAN variables which are directly related to the degrees of freedom and transformations are listed below.

NCDFA(I)	Number of DOF of component I

CDFLA(J,I)
CDFIA(J,I)
	Literal and integer parts of the name of the Jth DOF of component I

NCIDFA(I)	Number of implicit DOF associated with component I

CIDFLA(J,I)
CIDFIA(J,I)
	Literal and integer parts of the name of the Jth implicit DOF of component I

NCCOA(I)	Number of implicit coefficients associated with component I
CCOA(J,I)	Jth implicit coefficient of component I (Real)
CCODFA(J,I)	Corresponding component DOF
CINDA(J,I)	Starting index in CCOA, CCODFA for the Jth implicit DOF of component I
NSDF	Number of system DOF

SDFL(I)
SDFI(I)
	Literal and integer parts of the Ith system
	DOF name

TRAN(J,I)	Transformation constant for the Jth DOF of component I
NSCO	Number of system implicit coefficients

SCO(J)
SCODF(J)
	Jth system implicit coefficient (Real)
	Corresponding system DOF

DEFINITION PHASE

The mathematical basis of DYSCO resides primarily in "execute model" (Figure 6). Prior to entry to this execution module, "create new model" (Figure 5) which uses the C---I and F---I technical modules forms a "run data file" (RDF) which contains all the program control variables (see above): CORDA, CREFA, FORDA, FREFA, NUMA, CLUN, CREC, FLUN, and FREC. This information, along with the referenced files containing the specific inputs for the components and forces, is sufficient to completely define the coupled equations of the helicopter model (system equations).

In order to demonstrate the implementation, the example of Figure 3 with mechanical shakers on components 1 and 6 shall be used to follow the development of the creation of the model. "Create new model" will define the control variables as follows (omitting file references).

I	CORDA	CREFA	FORDA	FREFA	NUMA
1	1	1	3	1	1
2	2	1	0	0	2
3	1	2	0	0	3
4	1	3	0	0	4
5	2	2	0	0	5
6	1	4	3	2	6

Module 3.1.1 (Figure 7) will execute, in order, as specified by CORDA, the appropriate definition modules (3.1.1.1): CSF1D, CFM2D, CSF1D, CSF1D, CFM2D, and CFS1D. In each case, the module will read the appropriate input data from the RDF based on the logical unit and record numbers as given by CLUN and CREC. At the completion of the execution of the definition modules, the values of the component DOF variables described in the previous section will be as given below.

NCDFA = 3, 2, 5, 3, 4, 4

CDFLA, CDFIA*

COMPONENT	1	2	3	4	5	6
	X 1000	QFUS2100	Y 1000	Z 1000	QFUS5100	X 2000
	X 2000	QFUS2200	Y 2000	Y 2000	QFUS5200	Z 2000
	X 3000		Y 3000	Z 2000	QFUS5300	W 1000
			Y 4000		QFUS5400	W 2000
			Y 5000			

NCIDFA = 0, 3, 0, 0, 2, 0

CIDFLA, CIDFIA

COMPONENT	1	2	3	4	5	6
		Y 1000			Y 5000	
		X 3000			W 2000	
		Z 1000				

CCOA, CCODFA**

COMPONENT	1	2	3	4	5	6
I						
1		1.1, 1			2.5, 1	
2		- .3, -2			-1.2, 2	
3		.1, 1			3.5, 3	
4		.2, -2			- .1, -4	
5		-1.0, 1			-1.0, 1	
6		1.0, -2			-1.1, 2	
7					.2, 3	
8					2.0, -4	

*Note the standard DOF names for CFM2 components.
**Note that each implicit DOF series ends when CCODFA is negative.

CINDA

COMPONENT I	1	2	3	4	5	6
1		1			1	
2		3			5	
3		5				

The use of these DOF variables may be illustrated as follows: W 2000 (implicit variable 2 of component 5 as in CIDFLA, CIDFIA) is equal to a series starting in location 5 (CINDA (2,5)) of CCOA, CCODFA. The series is -1.0 * QFUS5100 -1.1 * QFUS5200 + .2 * QFUS5300 + 2.0 * QFUS5400, where the CCODFA values refer to the corresponding elements of CDFLA, CDFIA for the variable names.

The component DOF data formed, as described above, are then processed in 3.1.1.2 (Figure 7) to form the system DOF variables as defined in the previous section. These become

NSDF = 13

I	SDFL, SDFI
1	X 1000
2	X 2000
3	QFUS2100
4	QFUS2200
5	Y 2000
6	Y 3000
7	Y 4000
8	Z 2000
9	QFUS5100
10	QFUS5200
11	QFUS5300
12	QFUS5400
13	W 1000

where duplications and implicit DOF variables have been omitted. TRAN, SCO, and SCODF have been previously given after Figure 3. These are formed from NCDFA, CDFLA, CDFIA, NCIDFA, CIDFLA, CIDFIA, CCOA, CCODFA, and CINDA in subroutine XXCDT. The algorithm is straightforward and may be easily followed in the FORTRAN code.

TRAN, SCO, and SCODF is the only information needed to transform component coefficients and forces to system coefficients and forces and to retrieve component state vectors from system state vectors.

COEFFICIENT PHASE

The "compute constant coefficients" module (3.1.2), Figure 8, executes in the order specified by CORDA the C---C modules to compute the component constant coefficients and right-hand-side terms of the component equations (exclusive of force technical module computations). As each component's calculations are completed the subroutines XXCCM and XXCCF are used to transform this data to the data required in the coupled system equations. These algorithms have been discussed previously.

SOLUTION PHASE

The "execute solution" module (3.2), Figure 6, executes the specified solution technical modules, S---I, S---A, and S---O. The S---A module which performs the specified solution is somewhat different from and more complex in nature than the corresponding component modules. When this module is reached, the constant coefficients and forces of the differential equations of the system have been formed and the data have been supplied which will allow the solution of the equations. Several types of solutions are available to the user. It is presumed that the user has formulated a model which is appropriate to the type of solution specified.

The frequency domain solution (SFD1) and the eigenanalysis solution (SEA3) are purely mathematical operations on the system constant M, C, and K matrices and no further discussion is necessary.

The time history solution (STH3) involves solutions of the differential equations in the time domain. Figure 9 shows the hierarchy of the STH3 solution. To better illustrate the operation of this solution procedure, the modules of this figure will be briefly discussed. Typical solution input data (3.2.1) include integration increment, time to end of solution, initial conditions and various options. The output technical module (3.2.3) simply indicates that the solution is complete.

"Perform solution" (technical module STH3A) controls the following procedures, and tests for completion. "Compute $\dot{v}$, v" is the function of an integration method (such as Runge-Kutta) which takes the accelerations at a given time, t, and computes the velocities and displacements at the next time increment, t + Δt. The acceleration is obtained by solving $\ddot{v} = M^{-1}(F-C\dot{v}-Kv)$ at time, t. The values of $\dot{v}$ and v were obtained on the previous execution of the integration and M, F, C, and K are obtained from the previously obtained constant values and the execution of the active component modules (C---A).

The C---A module of each component of the particular helicopter model is executed which returns the component time varying M, C, K and F. These are then transformed and added to the system constant matrices by the same subroutines which were used in the "coefficient phase" (Figure 8). The force vector for each component is obtained by executing the user selected force active module (F---A). The calculation of nonlinear coefficients

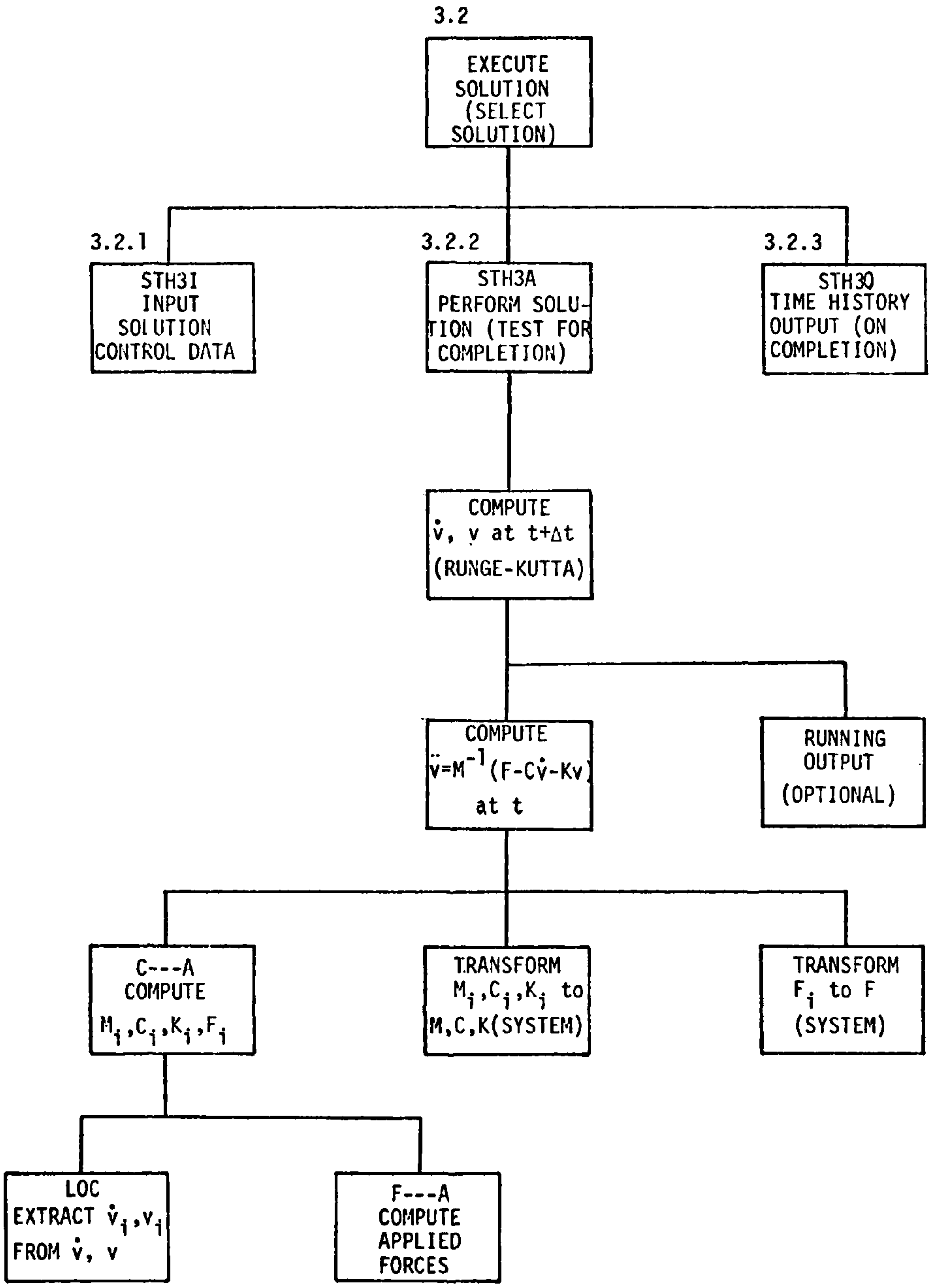

3.2
EXECUTE SOLUTION (SELECT SOLUTION)
3.2.1
STH3I INPUT SOLUTION CONTROL DATA
3.2.2
STH3A PERFORM SOLUTION (TEST FOR COMPLETION)
3.2.3
STH3O TIME HISTORY OUTPUT (ON COMPLETION)
COMPUTE $\dot{v}$, v at t+Δt (RUNGE-KUTTA)
COMPUTE $\ddot{v}=M^{-1}(F-C\dot{v}-Kv)$ at t
RUNNING OUTPUT (OPTIONAL)
C---A COMPUTE M_i,C_i,K_i,F_i
TRANSFORM M_i,C_i,K_i to M,C,K(SYSTEM)
TRANSFORM F_i to F (SYSTEM)
LOC EXTRACT $\dot{v}_i,v_i$ FROM $\dot{v}$, v
F---A COMPUTE APPLIED FORCES

Figure 9. STH3 Solution Hierarchy.

and forces requires information concerning local velocities and displacements of the component (and possibly other components). This information is obtained from the system state vector by use of the transformation data as previously described.

DATA MANAGEMENT

All component and force data, except special inputs required as solution data, are input in "create new model" (Figure 4). These data are entered using the C---I and F---I modules. All data required in "execute model" (Figure 4) are placed on a "run data file". There is no other data linkage between these two execution modules.

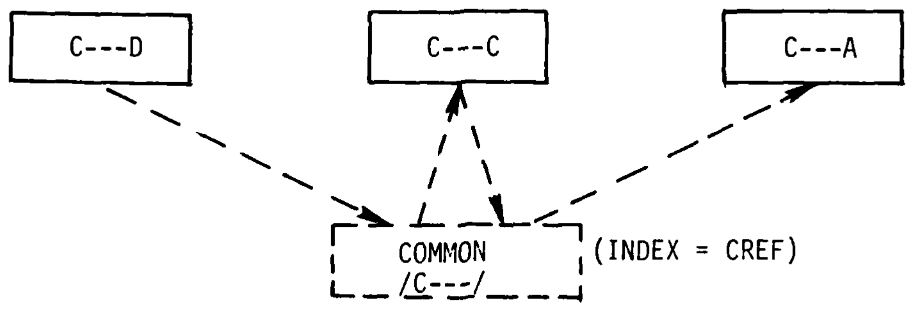
Within "execute model" (Figure 6) data are read from the run data file by the C---D, C---C, and F---C modules. These data for a given component are shared between the associated C---D, C---C, and C---A modules through a COMMON area named with the name of the component (C---). Similarly, the data for F---C and F---A modules are shared through a COMMON named with the name of the force representation (F---). In addition, a special COMMON/ROT/ is included in all rotor and control system component definition, coefficient, and active modules to share such information as the number of blades and rotational speed. The component named COMMONs contain information regarding the specific component degrees of freedom and physical parameters. These COMMON variables are dimensioned and indexed by the number of the reference to the component (CREFA) to distinguish data produced at each use of the component.

The force data is handled in a similar fashion. An additional COMMON area, /AFTAB/ is used to store aerodynamic tables.

Figures 10 and 11 illustrate the use of these COMMON areas. In these figures, CREF and FREF are the values of specific elements of CREFA and FREFA and NROT is a specific value of NUMA referring to a rotor component.

TECHNICAL MODULE SPECIFICATIONS

This section describes, for each of the technical modules, the input and output arguments, special COMMON areas, and the functions of the module. The argument lists for the same type of technical module are for the most part identical.



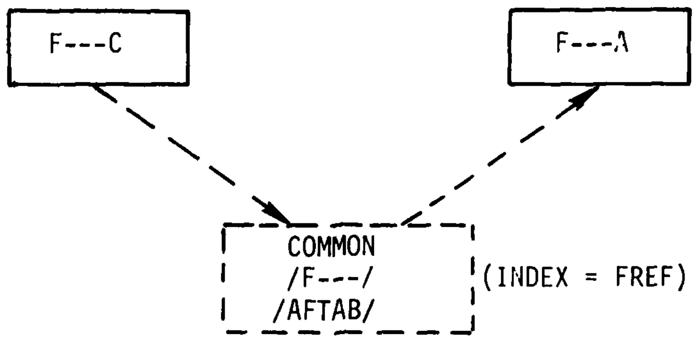


Figure 10. Component and Force Module COMMON Area Use.

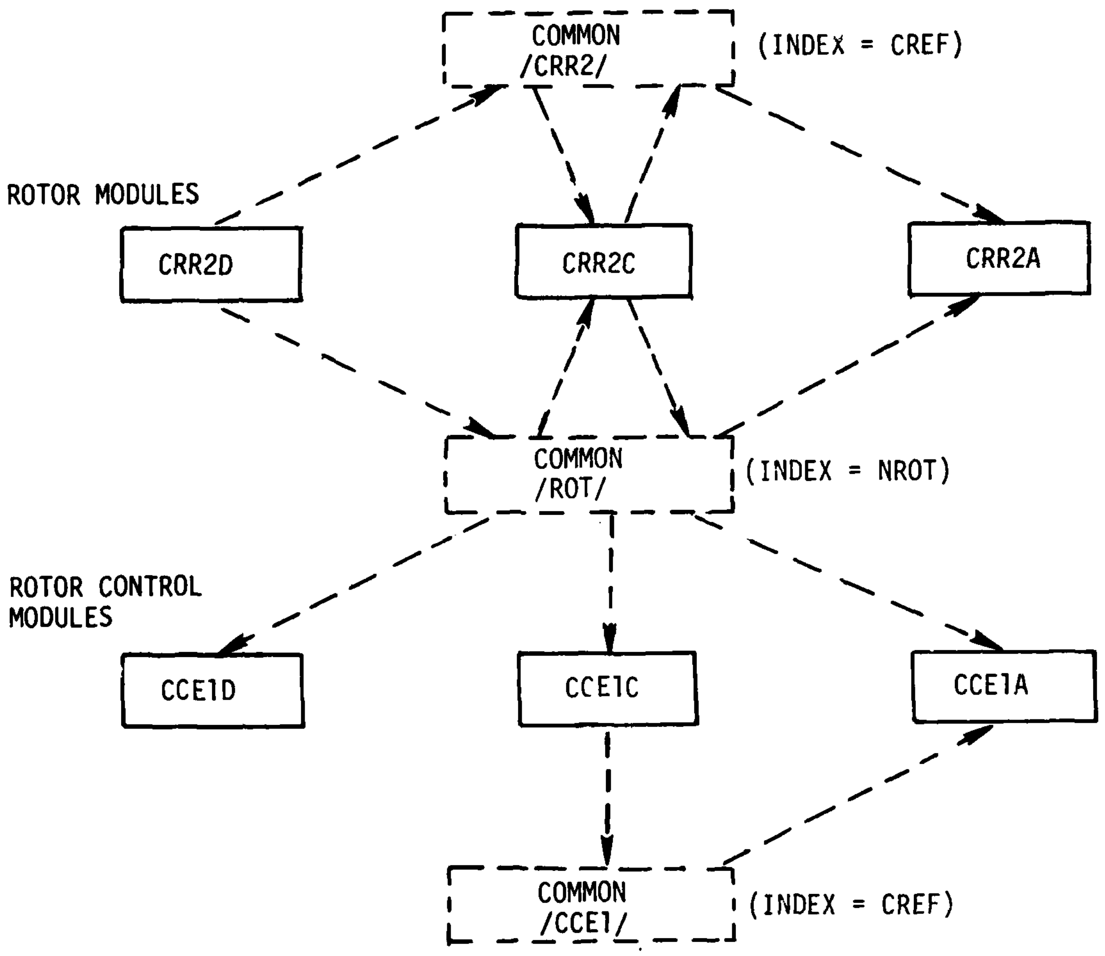


Figure 11. Rotor and Control System COMMON Area Use.

C---I, All Component Input Modules

INPUT ARGUMENTS

BATCH	-	indicator for batch or interactive input
IN, OUT	-	input, output unit numbers
RDF	-	unit number of run data file
DSNAME	-	(2), names of data sets for definition and coefficient data
NSTR(or NROT)	-	user assigned structure or rotor number

OUTPUT ARGUMENT

ERR	-	error indicator

COMMON AREAS

no technical COMMON area

FUNCTION

Accepts all necessary input and converts units and format and stores on RDF.

C---D, All Component Definition Modules

INPUT ARGUMENTS

NCDFX,NCIDFX,NCCOX	-	dimensioning parameters
IREF	-	specifies that this is the IREFth call of this module
NSTR(or NROT)	-	structure or rotor number
LUN, NREC	-	unit and record number for appropriate definition data of RDF

OUTPUT ARGUMENTS

NCDF	-	number of component degrees of freedom
CDFL,CDFI	-	arrays of literal and integer parts of the component DOF names
NCIDF	-	number of component implicit degrees of freedom
CIDFL,CIDFI	-	arrays of literal and integer parts of implicit DOF names
NCCO	-	number of implicit coefficients of component

CCO,CCODF	-	arrays of implicit coefficients and corresponding component DOF of the implicit relationships, packed
CIND	-	array of starting indices of coefficients in above for each implicit DOF

COMMON AREAS

/C---/	-	contains data shared with C---C and C---A. Data vary with component. All data in arrays subscripted by IREF. (Not used in CCE1D or CSF1D.)
/ROT/	-	for rotor and control system modules only. All data in arrays subscripted by NROT.
OM	-	rotational speed
NB	-	number of blades
PSIØ	-	reference azimuth at t = 0

Note: These data are formed in CRR2D and CRR2C and are used in control system modules and rotor force modules as well as CRR2C and CRR2A.

FUNCTION

Reads definition data from RDF and forms the DOF names and implicit DOF names and corresponding coefficients. Sets up information in COMMON areas, as required.

<u>C---C, All Component Coefficient Modules</u>

INPUT ARGUMENTS

NCDFX	-	dimensioning parameter
IREF	-	specifies that this is the IREFth call of this module
NSTR (or NROT)	-	structure or rotor number
NCDF	-	number of component DOF
LUN,NREC	-	unit and record number for appropriate coefficient data of RDF

OUTPUT ARGUMENTS

CM, ICHM CC, ICHC CK, ICHK CF, ICHF	-	component constant mass, damping, stiffness, and force matrices, each followed by indicator of null matrix

COMMON AREAS

/C---/	- see C---D discussion. Coefficients used in C---A may be formed in C---C or C---D.
/ROT/	- see C---D discussion

FUNCTION

Reads coefficient data from RDF and forms constant M,C,K and F matrices using data from RDF and COMMON areas. Also sets up information in COMMON areas.

C---A, All Component Active Modules

INPUT ARGUMENTS

NCX,NCDFX,NSDFX, NSCOX,NDIMX	- dimensioning parameters
ISEQ	- component number
NDIM	- number of DOF in first order formulation (2*NSDF)
IREF	- specifies that this is the IREFth usage of this module
NSTR (or NROT)	- structure (or rotor) number
FORD	- force module ordinal
FREF	- specifies that this is the FREFth usage of the specified force module
NCDF	- number of DOF of this component
NSDF	- number of system DOFs
TRAN,SCO,SCODF	- transformation data (see section on transformation methods)
T	- time
CV,CVDOT	- arrays of component DOF displacements and velocities. (These are internal variables, dimensioned externally.)
SV	- system state vector at time T, displacements and velocities

OUTPUT ARGUMENTS

CM,ICHM CC,ICHC CK,ICHK CF,ICHF	- component incremental mass, damping, stiffness, and force matrices (evaluated at time T and to be added to system constant matrices), each followed by indicator of null matrix

COMMON AREAS

same as C---C

FUNCTION

Forms component matrices at time T using CV,CVDOT, and force module.

F---I, All Force Input Modules

INPUT ARGUMENTS AND
OUTPUT ARGUMENTS

Same as C---I except for FRA2I which includes a run termination indicator, QUIT

COMMON AREAS

/AFTAB/	-	for FRA2I only, contains aerodynamic tables

FUNCTION

Accepts all necessary input and converts units and format and stores on RDF.

F---C, All Force Coefficient Modules

INPUT ARGUMENTS

NCX,NCDFX	-	dimensioning parameters
FREF	-	specifies that this is the FREFth usage of this force module
LUN,NREC LUN1,NREC1	-	units and record numbers for coefficient and tabular data(if used)
NC	-	number of components
ISEQ	-	component number
NCDFA	-	array of component number of DOFs
CDFLA, CDFIA	-	arrays of component literal and integer DOF names

Note: arguments NC to CDFIA used only when input refers to DOF names

OUTPUT ARGUMENTS

ERROR	-	for FRA2C only, error indicator
QUIT	-	for FRA2C only, termination indicator

COMMON AREAS

/F---/	-	contains data shared with F---A. Data vary with force module, subscripted by FREF.

/AFTAB/ - see F---I discussion

FUNCTION

Reads data from RDF and sets up information in COMMON areas.

F---A, Force Active Modules

INPUT AND OUTPUT ARGUMENTS

All arguments are specific to the type of force computed. Groups of force modules which are optionally accessed by a particular component module have the same argument lists.

COMMON AREAS

Same as in F---C except for COMMON/ROT/ (see C---D discussion)

FUNCTION

Computes applied forces for a given time and given component state vector(s)

Note: These are the only technical modules which are called by other technical modules (C---A).

S----, Solution Modules

In the present implementation the arguments are special to the particular solution method. In general, the S---I module accepts, validates and transforms the input data and places the necessary data in COMMON/S---/. The S---A module uses these data and performs the solution specified. For time domain solution S---A includes the "sequence control data" in its input argument list and will call C---A modules as appropriate. The S---O module outputs solution results at completion.

DYSCO HIERARCHY CHARTS

Figures 12 - 15 illustrate the actual hierarchy implemented in DYSCO. In addition, a number of the major subroutines and their functions are shown.

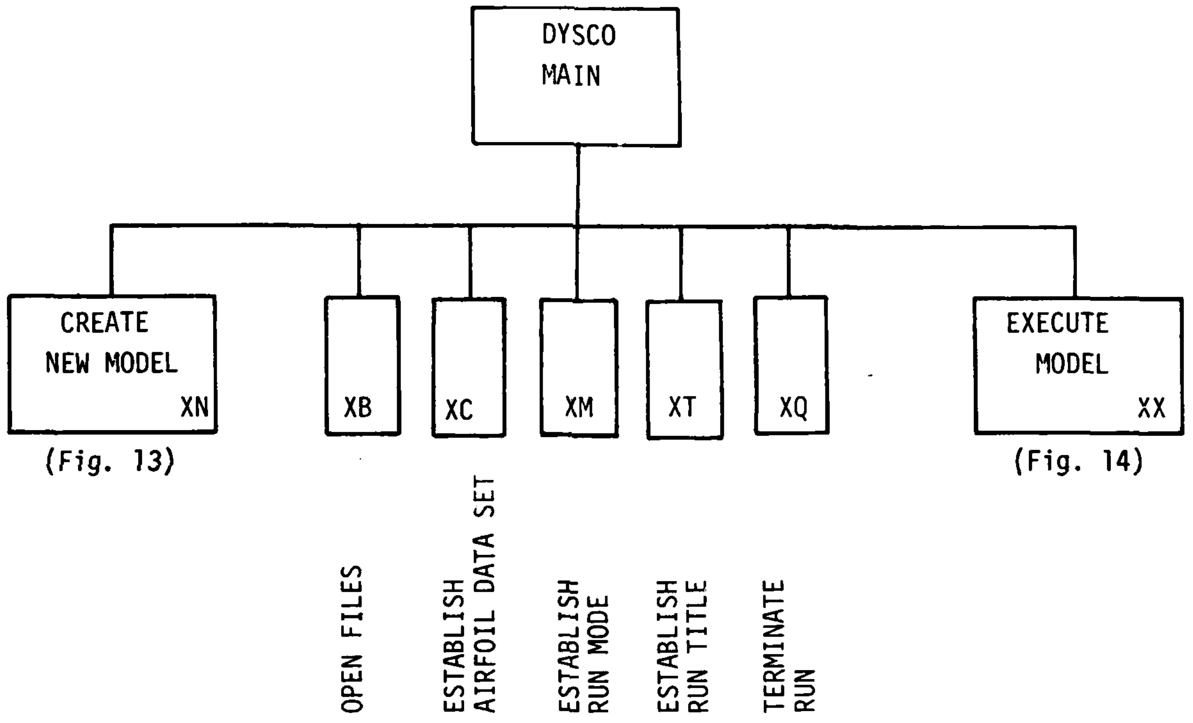


Figure 12. DYSCO Hierarchy, Top Level.

7
B

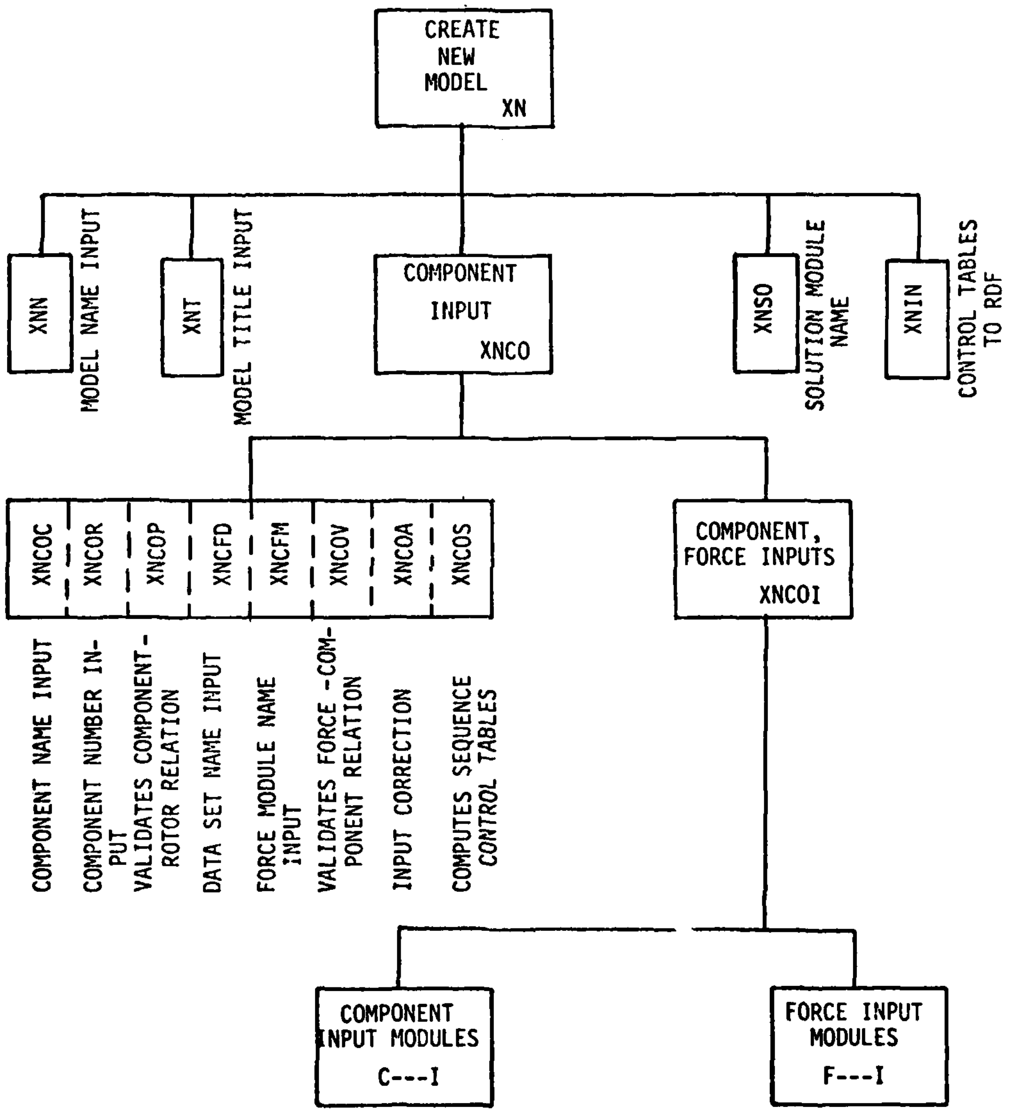


Figure 13. DYSCO Hierarchy, "Create New Model".

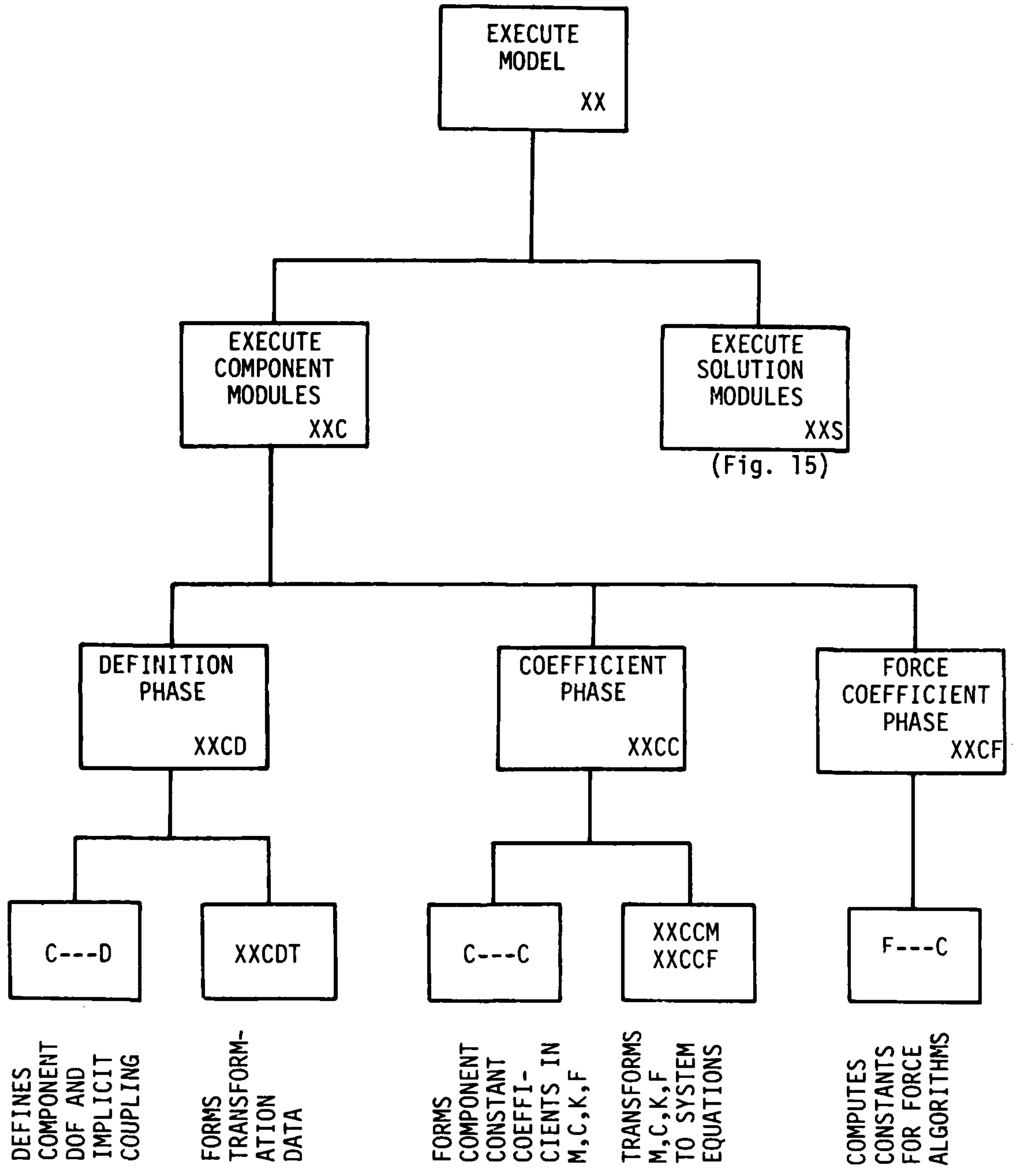

EXECUTE MODEL XX
EXECUTE COMPONENT MODULES XXC
EXECUTE SOLUTION MODULES XXS
(Fig. 15)
DEFINITION PHASE XXCD
COEFFICIENT PHASE XXCC
FORCE COEFFICIENT PHASE XXCF
C---D
XXCDT
C---C
XXCCM XXCCF
F---C
DEFINES COMPONENT DOF AND IMPLICIT COUPLING
FORMS TRANSFORM-ATION DATA
FORMS COMPONENT CONSTANT COEFFI-CIENTS IN M,C,K,F
TRANSFORMS M,C,K,F TO SYSTEM EQUATIONS
COMPUTES CONSTANTS FOR FORCE ALGORITHMS

Figure 14. DYSCO Hierarchy, "Execute Model".

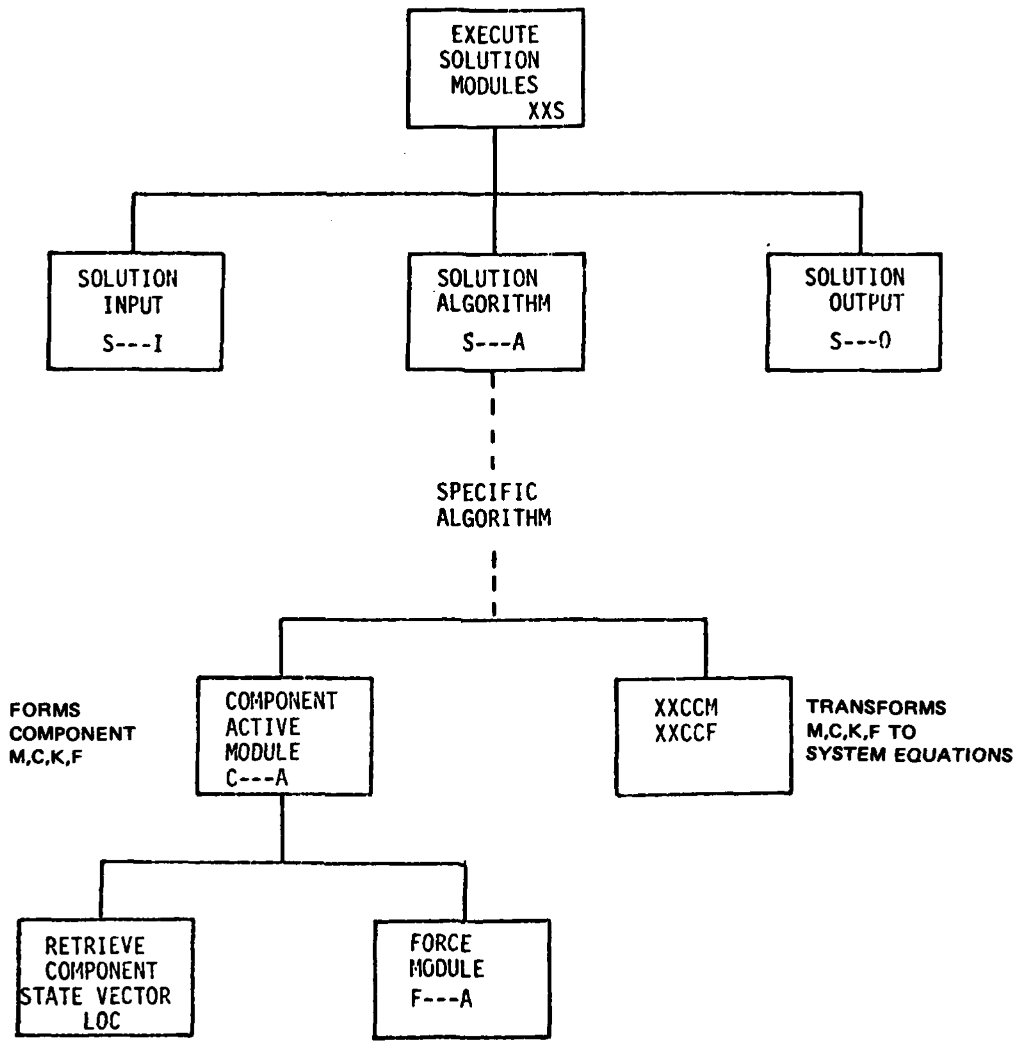


Figure 15. DYSCO Hierarchy, "Execute Solution Modules".

GENERAL APPLIED FORCE CAPABILITIES

One of the major concerns in rotorcraft analysis is the treatment of aerodynamic loads. In this section the general approach, the basic capabilities, and the limitations of the DYSCO design are discussed with regard to applied force representation. Included in the discussion are existing and possible implementations within the basic framework of the DYSCO program.

GENERAL COMMENTS

The types of applied forces relative to rotorcraft analyses fall into three categories*:

1. Mechanical vibratory forces as due to a shaker
2. Aerodynamic loads acting on nonrotating components
3. Aerodynamic loads acting on a rotor blade.

In each of these categories the forces often will be time dependent and/or nonlinear and dependent on the degrees of freedom of the system. Thus, the forces must be computed during a time history solution (as in Figure 9).

In a fully implemented DYSCO design, the user will have a choice of representations for any physical component and each component representation will generally have several force methods available to it. Also, each force method will generally be usable by several component technology modules. Different modules representing the same component may have different degrees of freedom (e.g., physical displacements or modal amplitudes) and corresponding different generalized forces. In any case, however, the physical displacements may be obtained from the degrees of freedom and the generalized forces may be obtained from the distributed forces. Thus, the general rule for all force computations (in F---A) is that physical displacements (and slopes, velocities as required) of distributed points on the structure be used as basic input and the applied forces (and moments) at these points be output.

The major steps in the process of computing applied forces in DYSCO are as follows (for each component at time, t, and given system state vector, as in Figure 9):

1. Retrieve component state vector (displacements and velocities of component degrees of freedom)
2. Convert to displacement, slope, velocity, angular velocity as required at each specified point on structure
3. Select specified force module
4. Compute applied forces and moments at each specified point on structure (in F---A)

*Steady forces such as gravity may simply appear on the right-hand side of the component equations, and are not discussed here.

5. Compute generalized forces acting on component
6. Convert component forces to system forces.

Step 1. is performed in subroutine LOC, steps 2., 3., and 5., are performed in the active component module, C---A, step 4. is carried out in the selected force module, F---A, and step 6. is performed in subroutine XXCCF.

In order to better illustrate this process and evaluate the approach taken in DYSCO examples and discussions of rotor blade loads analyses are presented in later sections.

DATA REQUIREMENTS FOR ROTOR BLADE LOADS

For levels of technology consistent with present practices, including nonuniform and dynamic wake calculation, there are three basic sets of data required to compute rotor blade airloads:

1. The physical position and motion of each point on the blade where loads are to be calculated
2. The aerodynamic environment at each point on the blade including all components of the fluid velocity
3. The aerodynamic characteristics of each point on the blade, i.e., the data necessary to convert 1. and 2. above into forces and moments.

One of the features of the DYSCO design is the separation of component representations to allow arbitrary use of any combination of components in any particular analysis. This requires the separation of data which describes each component to the greatest extent possible. The same considerations apply to the data associated with the force computations.

The source of the force computation data must be examined and where options exist (as to source) the choice must be made which is the most logical from the point of view of the user. The sources presently available in DYSCO are:

COMMON/C---/ COMMON/ROT/	(from component input)
COMMON/F---/ COMMON/AFTAB/	(from force input)
Component state vector (from system state vector)	

The above basic sets of data are now considered in more detail and some of the technical considerations are discussed.

Blade Position and Motion

When the loads are computed on the basis of airfoil characteristics the blade position and motion are required in order to compute the local angle of attack and its rate of change.

The blade stations at which the loads are to be computed may be part of the component input or the force input. Their definition is most convenient by component input, since the blade station data are also required for the computation of the generalized forces and define the stations at which modal displacements are given if a blade modal representation is used (or the component degrees of freedom are located if a finite element representation is used). This station information is in COMMON/C---/ in the present implementation of DYSCO. When a modal representation is used, the mode shapes and modal slopes must be available through COMMON/C---/.

The above information plus the displacement and velocities of the component degrees of freedom are required to obtain the position and motion of each station. These data are obtained from the component state vector.

Additional information required, such as number of blades, rotational speed, and reference azimuth angle, is found in COMMON/ROT/ in the present version of DYSCO.

Aerodynamic Environment

The blade position and motion in its local coordinate system must be transformed to some global reference axis system, such as wind axes. Inputs which define the orientation of the shaft axis to the reference axis are quite appropriate in the force module input. This information is in COMMON/F---/ for each rotor.

The wind velocity in the reference system along with parameters such as altitude, density, and sound velocity is input in the force module input and this information is also in COMMON/F---/.

If the induced velocity is constant or prescribed, this information will also be in COMMON/F---/ either through direct input or by computation in F---C.

If the induced velocity is computed by an iteration process or if the effects of other components are involved, a more complex procedure is required. These conditions are discussed in a later section.

Aerodynamic Characteristics

The aerodynamic characteristics include geometrical parameters such as chord, ac-cg offset, and airfoil section. These data are most logically supplied as force input* and are in COMMON/F---/ since they do not have an effect except in the computation of the aerodynamic forces. Tables of aerodynamic coefficients will be accessed from a data bank through input to the force module and placed in COMMON/AFTAB/.

SIMPLE CASE

A simple case of rotor blade force computation is illustrated in Figure 16. This case would include the following capabilities if appropriate component modules are implemented: modal or finite element blade representation, linear or tabular aerodynamics, and uniform or prescribed inflow.

By the use of an appropriate force algorithm, this illustration includes all cases where the airflow relative to the rotor coordinate system is not affected by other components or by the load distribution on the rotor.

The figure also shows the sources of the data used. The manner of treating two special cases of blade representation are described below.

Segmented Blade

Consider a simplified finite element representation of a blade which is made up of rigid segments connected by equivalent springs and where each segment may have up to six degrees of freedom.

The component technology module may be designed so that the aerodynamic stations are independent of the physical segments or they may be the end points of the segments or there may be several aerodynamic stations per segment. Decisions of this type are up to the designer of the component technical modules. In any of the above definitions of the aerodynamic stations, given the values of the degrees of freedom of the component, it is possible to compute the data required for input to the force module. The degrees of freedom are the hub motions with respect to the shaft axis and all the degrees of freedom of each blade segment. The computation of the position and motion of each aerodynamic station is routine.

The position and motion of each aerodynamic station is passed to the force module, F---A, which returns the forces and moments per unit length at each aerodynamic station.

These forces and moments are transformed into the generalized forces by computing the effective forces and moments acting at each degree of freedom of each segment.

*The present implementation requires the chord to be input in the rotor component module for convenience in treating nonuniform blade data.

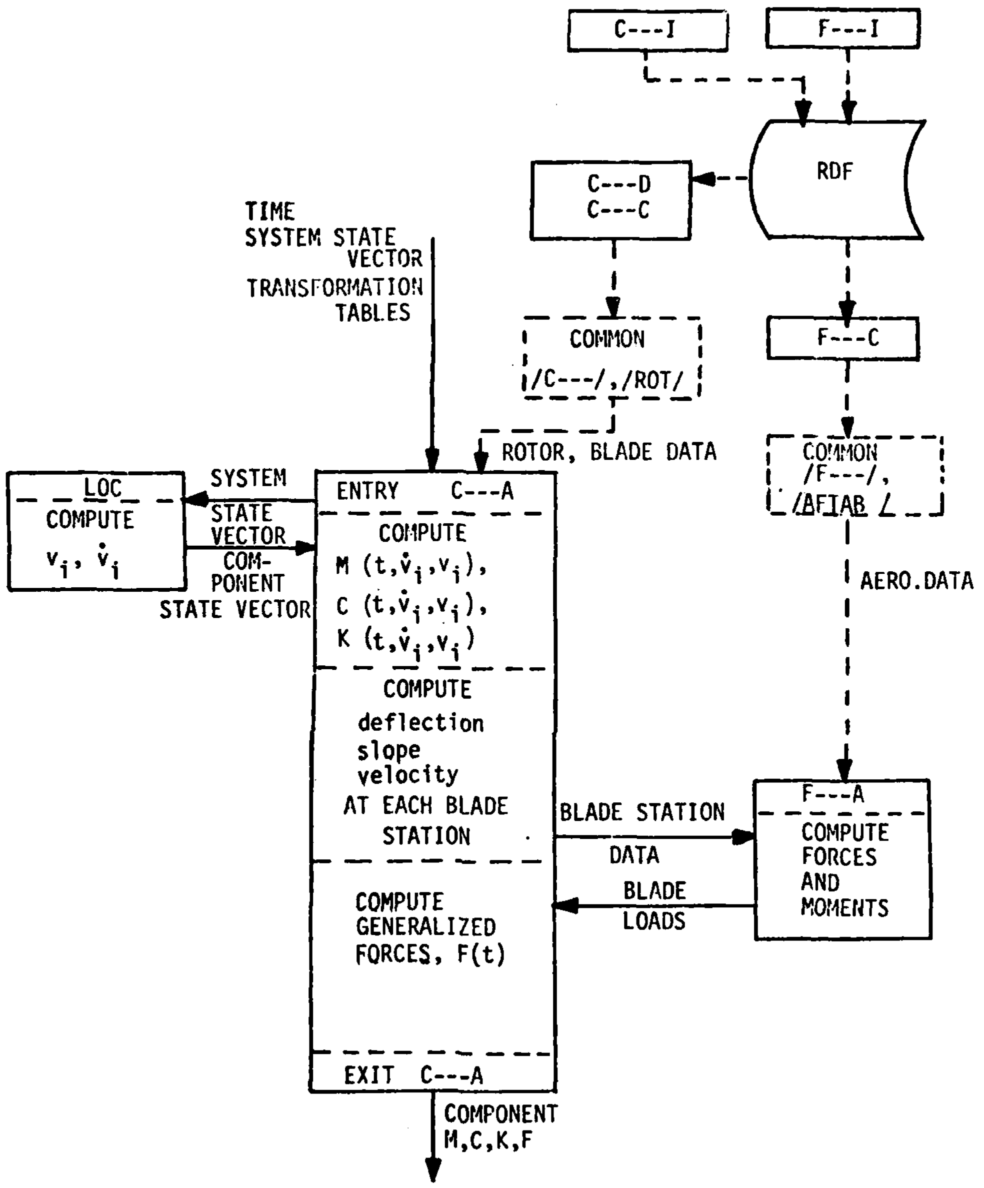


Figure 16. Flowchart Illustrating Rotor Force Computation.

Modal Blade

Probably the mcre common representation of a blade is that using modal coordinates (rigid blade, uncoupled elastic, or fully coupled modes).

In this case the degrees of freedom are the modal amplitudes. The displacements, slopes, and velocities in the in-plane, out-of-plane, and torsion directions at each station are computed simply by multiplying the appropriate components of the modes by the appropriate degrees of freedom and adding the effect of the hub motion.

The input to the force module is identical in form to that of the segmented blade example, abcve. The force module also returns the identical information as above.

The generalized forces are formed by multiplying the forces by the modes and integrating. Note that the quite different blade representations of this and the previous section may use the identical force computation module.

LIMITATIONS IN CAPABILITIES OF FORCE MODULE

The capabilities of the F---A module (e.g., in Figure 16) are limited only by the information available to it and, of course, by the technology of converting the input to the output.

In the example shown in Figure 16, the F---A module is entered at each time point during the integration of the system equations. Depending on the detailed design of C---A and F---A, the force module may be called for each blade station, for each blade, or for all blades on the rotor simultaneously. In the present DYSCO program, one blade at a time is analyzed, thus at each point in time F---A is called as many times as there are blades on the rotor. This could have just as conveniently been written to treat all the blades at one time.

The information available through C---A is:

all blade physical parameters, time,
blade azimuths, rotor degrees of freedom,
position and velocity of each blade station

The information available through COMMONs associated with the force module is:

relationship of rotor coordinates to global axis system, wind direction and velocity, fluid properties, aerodynamic characteristics of blade (airfoil sections, force and moment coefficient or tables), and induced velocity distribution (uniform or prescribed)

The simplest use of this information is to compute the relevant wind velocities, the angles of attack, and the rates of change of angle of attack, and convert these into forces and moments by use of the aerodynamic coefficients. Note that the prescribed wake may include the approximate effects of blade interactions and body interactions. The effects of blade warpage may also be included if appropriate degrees of freedom are used in the model.

Without changing the input or output arguments to F---A in the CALL statements in C---A, it is possible to include certain other effects as discussed below.

Improved Wake Model

The force module could be designed to store an accumulated history of blade motions and load distribution and use this information to compute or correct the induced velocity distribution. This capability is within the design shown in Figure 16 (and Figure 9).

A more complex model which is still consistent with the DYSCO design is to compute an induced wake by an iterative procedure: (1) assume (or compute) a distribution of inflow; (2) compute a rotor revolution (or continue to steady state), and (3) use this data to recompute the inflow map and return to (2) until convergence. This procedure would require modifications to the chart of Figure 9 and would involve a new module to correct the inflow field.

Rotor-Body Interference

Three levels of complexity of rotor-body interference may be considered. The simplest level is when the interference effect is based only on the physical relationships of the structures. This level may be treated within the design shown in the chart of Figure 16.

The second level is when the interference effect is a function of the body motion or deformation. Determination of this effect requires information regarding a structure other than the rotor. This information could be obtained if F---A were to call subroutine LOC to obtain the state vector of the body. Information would be required (from COMMON/F---/) which identifies the body as a component of the system and identifies the degrees of freedom required. Such a capability would be a straightforward modification of the F---A module in Figure 16. Changes in input from C---A are not required; thus this capability falls within the scope of Figure 16.

The third level of complexity would involve the effects of the aerodynamic load distribution of the body. Such an analysis could possibly be designed into the chart shown. An iterative solution would require changes as discussed for the wake in the previous section.

Rotor-Rotor Interference

This involves the same considerations as above.

SUMMARY OF CAPABILITIES

In general, whenever aerodynamic loads are to be computed using only information about the component itself and on present (or past) state vectors, the required analysis is within the capabilities of the chart of Figure 16.

If, in addition, the motion of other components influences the flow field, this effect can also be included within the Figure 16 capabilities, if a call of subroutine LOC is initiated in F---A. The identification of the other components involved and the degrees of freedom required would have to be passed through COMMON/F---/.

When an iteration procedure is required, changes or additions to the solution algorithm may be required.

Analyses in which the loads on one component are dependent on the loads on another component and require the simultaneous computation of these loads will require further consideration.

APPLICATION CAPABILITIES OF DYSCO DESIGN

In order to demonstrate the capabilities of the present DYSCO design to treat a variety of typical rotorcraft problems an extended library of technology modules is hypothesized and a brief description of their use for various problems is presented.

The problems treated here are all solved in the time domain and involve periodic and nonlinear effects. Simpler problems could be solved with single special-purpose modules on a preprocessing level. These would be especially useful in the areas of preliminary trim analyses and for some stability and control analyses. These have not been illustrated.

In the illustrated problems, it is assumed that all necessary physical data and option data are available for each of the modules. Notes highlight some of the key options to be selected.

HYPOTHETICAL LIBRARY OF MODULES

A library of component, force, solution, and postprocessing modules for a hypothetical DYSCO-like program which is adequate to treat the problems illustrated is given below.

Modules having names identical to those implemented are assumed to be extended versions. All physical components have a gravity force option. Internal load option refers to the optional computation of bending moments, internal shears, and stresses which are written to an output file for postprocessing.

Component Technology Modules

CRAØ	-	Rotor, actuator disk, up to 6 hub DOF.
CRR2	-	Rotor, rigid hinged blades, 6 hub DOF and 3 DOF per blade (all optional). Spring, dampers, coupling, nonuniform blade properties.
CRM4	-	Rotor, modal blade representations, 6 hub DOF plus blade mode DOFs. Unequal blade properties and spacing allowed. Internal load option.
CRF6	-	Rotor, finite element blade and hub representations. Internal load option.
CSF1	-	Structure, general finite elements, all linear, arbitrary DOFs. M,C,K input.
CFM2	-	Fuselage, modal (including rigid body), rotor and other structure interfaces. Internal load option.

CFF4	-	Fuselage, finite element representation, M,C,K with rotor and other structure interfaces. Internal load option.
CCE1	-	Control system, single rotor, swashplate with elastic control rods (may be used with CRR2, CRM4, CRF6).
CCE4	-	Control system, multiple rotors, cockpit controls (stick and pedal displacements) coupled to rotating system and coupled to rotor.
CPD4	-	Pilot, dynamic model. Couples to any DOFs of CFM2, CFF4, CSF1 and cockpit controls of CCE4.
CPA6	-	Pilot, adaptive controls. Responds to any specified DOF, adjusts cockpit controls of CCE4 to achieve specified responses.

Force Technology Modules

FROØ	-	Rotor forces and moments, special for CRAØ.
FRAØ	-	Rotor blade forces, linear aerodynamics, blade pitch controls option. May be used with all CR-- except CRAØ.
FRA2	-	Rotor blade forces, tabular aerodynamics, prescribed inflow map. May be used with all CR-- except CRAØ.
FRF4	-	Rotor blade forces, unsteady, special module for flutter analysis (including stall flutter), prescribed wake. May be used with all CR-- except CRAØ.
FFAØ	-	Fuselage aerodynamics, flat plate drag only. May be used with all CF--.
FFA2	-	Fuselage linear or tabular aerodynamics including forces and moments and includes fixed aerodynamic surfaces.

Solution Technology Modules

STH3	-	Time history, initial conditions may be from restart file, optional error checks, termination conditions: time, variable tests, periodicity.
STR2	-	Trim (by iteration), uses STH3 to solve for steady state, then varies controls to achieve specified condition. Optional tests of trim, with tolerances selected by user.
SSF2	-	Stability analysis using Floquet analysis. Uses STH3 to obtain Floquet matrix, then performs eigenanalysis.
SSD3	-	Stability derivative analysis, automatic perturbation of controls, analysis of results. Uses STH3.

Postprocessing Technology Modules

PHA1	-	Harmonic analysis of specified file of solution data.
PMB4	-	Moving block FFT of specified file of solution data.

TRIM

A simple, predesign level of complexity, trim problem may be formulated as follows. In this example a four-bladed rotor on a rigid fuselage is modeled. Gravity forces are included in the component modules and blade control inputs are applied and varied in FRAØ. The vehicle is trimed in vertical, longitudinal, and pitch directions.

TRIM - SIMPLE

Component	Force	Notes
CRR2		Main rotor, Hub DOF = vertical, longitudinal, pitch, 4 blades with flapping (7 DOF)
	FRAØ	Linear aerodynamics, blade pitch control
CFM2		Fuselage, 3 rigid body modes: vertical, longitudinal and pitch
	FFA2	Linear aerodynamic, fuselage drag, horizontal tail lift
Solution		
STR2		Iterate to trim, 7 system DOF (3 fuselage, 4 blade flapping)

A problem of a higher level of complexity with all six fuselage rigid body degrees of freedom being trimmed could be formulated as follows.

TRIM - MORE COMPLEX

Component	Force	Notes
CRR2		Main rotor, 6 hub DOF, 4 blades with flapping and lag DOF (14 DOF)
	FRAØ	Linear aerodynamic, blade pitch control
CRAØ		Tail rotor, 6 DOF
	FROØ	Linear aerodynamics, rotor forces and moments
CFM2		Fuselage, 6 rigid body modes, 1 elastic vertical, 1 elastic lateral modes (8 DOF)
	FFA2	Linear aerodynamics, fuselage drag, horizontal and vertical tail surfaces
Solution		
STR2		Iterate to trim, 16 system DOF (8 fuselage, 8 main rotor blade)

TRANSIENT RESPONSE

Any model may be solved for transient response. To obtain the transient response of a trimmed condition, simply follow the trim by STH3 as shown below. In this example, the DOFs are perturbed. Similar approaches could perturb controls or the wind velocity or direction.

PERTURBED TRANSIENT RESPONSE

Component	Force	Notes
CRR2	FRAØ	See TRIM-MORE COMPLEX
CRAØ	FROØ	
CFM2	FFA2	
Solution		
STR2		Iterate to trim
STH3		Time history. Initial conditions from STR2 plus perturbation of single DOF
STH3		Repeat as above perturbing other DOFs

MANEUVER

A prescribed maneuver, in which a particular flight path time history is achieved is illustrated in this example. An adaptive pilot module is used to control the vehicle. In any particular analysis, it may not be possible to exactly achieve the specified maneuver. Such a procedure would include "fly-to-trim" as well as NOE capabilities. A somewhat more complex analytical model than was used in previous examples is illustrated below.

PRESCRIBED MANEUVER

Component	Force	Notes
CRM4		Main rotor, 6 hub DOF, 5 blades with flap, lag, pitch DOF, and two out of plane and one in plane elastic modes (36 DOF)
	FRA2	Tabular aerodynamics
CRR2		Tail rotor, 6 hub DOF, 3 blades with flapping DOF (9 DOF)
	FRAØ	Linear aerodynamics
CFM2		Fuselage, 6 rigid body and 6 elastic modes (12 DOF)
	FFA2	Fuselage, tabular aerodynamics and fixed horizontal and vertical tail
CCE4		Control system for both rotors (8 DOF, elastic control rods)
CPA6		Adaptive pilot
Solution		
STR2		Trim vehicle (45 DOF in model, trimmed on 6 DOF)
STH3		Time history of maneuver

AEROELASTIC STABILITY

An aeroelastic stability analysis of a simple two-bladed rotor on a simple whirl stand may be carried out as follows.

AEROELASTIC STABILITY - SIMPLE

Component	Force	Notes
CRR2		Rotor, 6 hub DOF, 2 blades with flap and lag DOF (10 DOF)
	FRAØ	Linear aerodynamics
CSF1		Whirl stand, C, K matrices representing springs and dampers from shaft to ground (6 DOF)
Solution		
SSF2		Floquet analysis (10 DOF)

Or alternatively:

Solution		
STH3		Time history from perturbation of selected DOF
Post Processing		
PMB4		Moving block FFT of time histories

WHIRL FLUTTER

Whirl flutter is included in the phenomena represented in the previous section on aeroelastic stability. A more complex problem of a stiff rotor (or propeller) on an elastic pylon connected to a flexible structure (wing) may be modeled as follows.

WHIRL FLUTTER

Component	Force	Notes
CRF6		Hingeless rotor, numerous finite element DOF plus 2 hub angular DOF and 2 hub translational DOF
	FRAØ	Linear aerodynamics
CSF1		Pylon, represented by simple (8 DOF) model linking rotor to wing
CFM2		Wing, cantilevered with 3 flatwise and 2 torsion modes
Solution		
STH3		Time history
Postprocessing		
PMB4		Moving block FFT

STALL FLUTTER

A typical stall flutter analysis may be performed on an isolated rotor in a prescribed wake as shown below.

STALL FLUTTER		
Component	Force	Notes
CRR2		Rotor without hub DOF, blades with flap, lag, pitch DOF
	FRF4	Unsteady loads analysis
Solution		
STH3		Time history
Postprocessing		
PMB4		Moving block FFT

PILOT COUPLED OSCILLATIONS

A simple example of pilot coupled oscillations for a tandem helicopter in vertical and pitch degrees of freedom may be carried out as shown below.

PILOT COUPLED OSCILLATIONS - TANDEM HELICOPTER

Component	Force	Notes
CRR2		Fwd rotor: vertical, pitch, longitudinal hub DOF, flapping and pitch DOF for each blade (11 DOF, 4 blades)
	FRAØ	Linear aerodynamics
CRR2		Aft rotor, same as forward rotor
	FRAØ	
CFM2		Fuselage, vertical and pitch rigid body modes, 2 vertical elastic modes (4 DOF)
CCE4		Control system for 2 rotors (8 DOF, elastic rods)
CPD4		Dynamic pilot, couples fuselage DOF to control system, approx. 12 DOF of which 6 are coupled to CFM2, CCE4
Solution		
STH3		Time history (may be preceded by STR2, Trim) (26 DOF)
Postprocessing		
PMB4		Moving block FFT

GROUND AND AIR RESONANCE

Simple ground resonance is adequately treated by the components and solution methods already described using component modules CRR2 (or CRM4 or CRF6 for a hingeless rotor) with lag DOF and no aerodynamics plus CSF1 (or CFM2 with CSF1 to simulate landing gear characteristics). It is also possible to design a special module to represent nonlinear landing gear characteristics. The effects of partial airborne conditions (for nonlinear landing gear) can be approximated (using CSF1) by varying the damping and spring rate inputs.

The stability analysis may be performed by either of the two techniques previously described.

Air resonance is similar to ground resonance except that rotor aerodynamics (linear) and blade flapping DOF should also be included along with a rigid body fuselage (CFM2) but without landing gear. Trim may be carried out prior to the analysis.

STABILITY AND CONTROL

Control positions versus flight condition are obtained from trim analyses of the configuration under study. At each trim point execution of SSD3 will yield the stability derivatives by perturbing the controls and carrying out and analyzing a time history solution.

INTERNAL LOADS AND VIBRATIONS

Internal loads may be determined in all the above problems, provided the component representations used are adequate for this purpose. A solution run for the purpose of determining internal blade loads (such as bending moments) must use at least CRM4 or CRF6. In the case of CRM4 an adequate number of modes must be used to provide the accuracy required. These loads may be harmonically analyzed or subjected to an FFT analysis.

For vibration analysis, the representations of the components may require less detail than for internal loads. For the study of vibration absorbers (rotating, nonrotating, or remote) a simulation may be obtained using CSF1 as long as the device is linear. Otherwise special modules may be easily developed.

For the study of vibration isolation systems it may be appropriate to use two CFM2 or CFF4 modules separated by CSF1 modules representing the equations of the isolators.

DYSCO BASED TECHNOLOGY COMPLEX

GENERAL DISCUSSION

The DYSCO program design and implementation is an outgrowth of a predesign study of a comprehensive helicopter analysis system commonly referred to as 2GCHAS[2]. Recent plans for implementation of such a system divide the functions into an "Executive Complex" and a "Technology Complex".

In this section the technical approach of DYSCO is used as a basis for the design of a Technology Complex. Some of the significant attributes of such a system and suggested approaches are described below.

Component Definition

In DYSCO a "component" may be any part of a vehicle which is represented by a second-order or lower differential equation in time. A component may be an entire helicopter, a rotor, a blade, or a balance weight. The equations of the component may contain periodic terms or any nonlinearities involving velocity or displacement of the degrees of freedom of the component (See MATHEMATICAL BASIS OF DYSCO, above). The degrees of freedom may be physical displacements, modal displacements, or any generalized degrees of freedom which can be coupled with the degrees of freedom of other components to result in a meaningful model of a rotorcraft.

The specific components which are to be modeled should not have an impact on the design of either the Executive Complex or the "execut ve routines" of the Technology Complex. The decisions as to which components are to be modeled should be made during the detailed design of the Technology Complex for the various system releases. No decisions, however, should limit the kinds of component representations which can be added to the system in the future.

Force Computation

Within the scope of the definition of the equations of a component, the applied forces could often be considered to be terms of the equations. It is, however, desirable to allow the user the capability to select force algorithms of various levels of complexity for the same physical representation of a component. Therefore, because of this consideration and because more advanced force representations may depend on the degrees of freedom of components other than the one being forced, the force computation should be separated from the component representation.

2. "Predesign of the Second-Generation Comprehensive Helicopter Analysis System", Control Data Corporation, USARTL-TR-78-43, Applied Technology Laboratory, AVRADCOM, Fort Eustus, VA. October 1978, ADA064131.

Input Requirements

One of the basic guidelines for the development of DYSCO was that a selection of various levels of complexity of analysis be available for each major component. For example, rotor representations may start with a simple actuator disk without separate blade degrees of freedom, proceed to a rigid hinged blade analysis, a modal blade analysis, and a highly nonlinear and detailed finite element representation of the blade.

The concept of a single set of descriptive data for a given rotor which can be used by any of the available rotor representations may be impractical and inefficient when less than the most complex method is to be used.

In DYSCO each component representation has its own input module which is tailored to the requirements of the particular analysis. Certainly, standards should be specified to make the input as uniform and consistent as possible. This is especially important when a component of a particular rotorcraft is to be modeled at more than one level of complexity.

The concept of a master data base (MDB) containing global information related to specific vehicles should be considered. In this case, the input modules for each component would extract the specific data required from the data base.

Separation of Executive and Technology Functions

The Executive Complex may be defined as that portion of the analysis system which controls the computer operations, such as data input/output, master data base control, module loading, and user language implementation. The Technology Complex is defined as that portion of the analysis system which includes the modules that contain the various mathematical models of the physical system and the modules for analyzing these models.

In DYSCO the Executive Complex may be defined as that portion of the program which maintains the data bases, accepts input which defines the model to be analyzed and operations to be performed, and establishes a run data file. The input which defines the model includes the names of the component and force modules and the specific inputs required by the corresponding input modules. The run data file consists of information regarding the sequence of operations and the specific data required by each module. The Executive Complex must have access to a list of valid component, force, and mathematical operation modules as well as all the input modules.

Also, in DYSCO the Technology Complex may be defined as that portion of the program which forms the equations based on the sequence specified in the run data file and carries out the mathematical operations specified.

A number of additional functions which are independent of the specific technical modules include sequential execution of modules, formation of transformation tables, and coupling of equations.

User Language

In order that the capabilities of the analysis system maintain a high degree of flexibility, a user language which is independent of the technical modules but controls their functioning seems to be inappropriate. Each technical module should have its own set of required inputs and options and should prompt the user in an interactive mode or have a specific interface with a master data base.

OVERVIEW

An overview of a complete analysis system based on the DYSCO program is illustrated in Figure 17.

The functions of the Executive Complex are:

1. Accept name (identifier) of model (to be formed).
2. Accept names of component modules and force modules of model and solution and pre- and postprocessing modules.
3. Access input modules of 2., accept input or a reference to data on file (of MDB) and write properly formatted data to RDF and optionally to MDB for future reference.
4. Check all input 2. for validity and compatibility of force and component modules.
5. Form tables of data to specify order of execution of technical modules, usage number, record number, and other information necessary to control execution of Technology Complex. These tables are placed on the RDF.
6. Store complete RDF on the MDB which becomes a PFC (particular functional capability), for future use.
7. Print lists of available modules, descriptions, tutorials as requested by user.
8. Edit any files on MDB.
9. Add new technical modules and update internal tables used in steps 4. and 5.

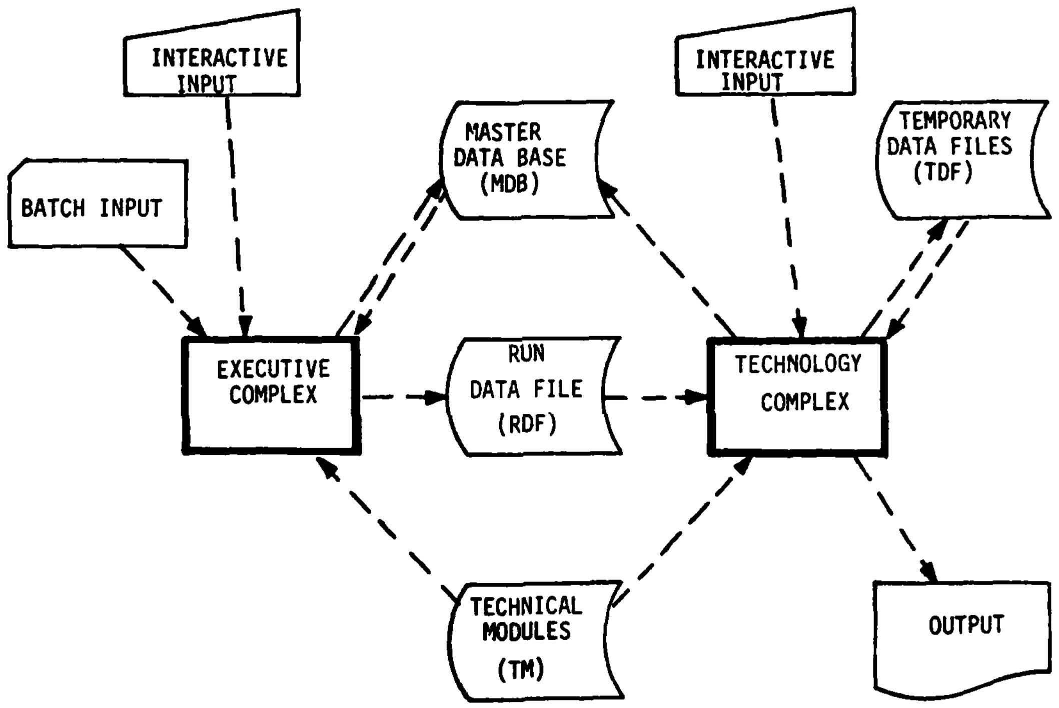


Figure 17. Overview of Analysis System Based on DYSCO.

The MDB contains:

1. Public and private files
2. Complete problem descriptions (PFCs)
3. Component data for re-use by technical modules
4. Data developed by preprocessing for future use by other technical modules
5. Standard tables such as airfoil characteristics

The Technology Complex is discussed below but several observations are appropriate at this point.

1. The information in the RDF plus the appropriate technology modules is a complete description of the model and the analysis and is adequate to complete all the required computations.
2. The interactive input is optionally used during the solution phase to modify parameters, to restart, or to initiate a new solution.
3. The temporary data files are used to store solution data for postprocessing analyses and to store preprocessed data for use by other technical modules.

TOP LEVEL TECHNOLOGY COMPLEX DESIGN

The design presented here differs from that of Figure 4 because of the separation into the Executive Complex and Technology Complex. The "create new model" of Figure 4 has become part of the Executive Complex. The major differences, however, are in the manner of presentation. The technical approach of DYSCO has been virtually unchanged.

A top level hierarchy diagram is shown in Figure 18. As can be seen the three major processes performed are preprocessing, coupled analysis, and postprocessing.

Preprocessing is defined as operations performed by specific technical modules, not involving the coupling of separate components. The generated data are normally for use by the coupled analysis to follow. Examples are uncoupled rotor blade mode calculations, rigid wake calculations, and preliminary trim analysis.

The coupled analysis includes defining the system in terms of the independent degrees of freedom of the system, the formation of the coordinate transformation data, the formation of the equations of the system, and the solution of these equations.

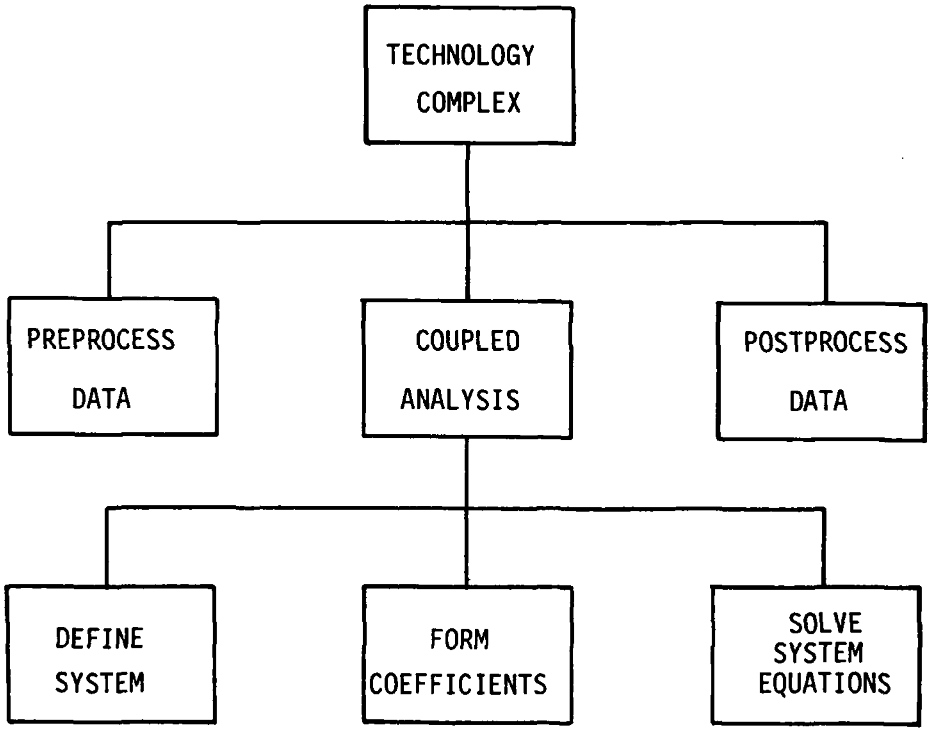


Figure 18. Top Level Hierarchy, Technology Complex.

Postprocessing is the analysis of the data normally obtained from a coupled analysis. Examples are harmonic analysis of blade internal loads, acoustic field calculations, stability analysis of time history data.

BASIC FUNCTIONAL CHARACTERISTICS

In reference to Figure 17, it is seen that the only data connection from the Executive Complex to the Technology Complex is through the run data file (RDF). Once the RDF has been completed there is no further need for the Executive Complex to remain in main storage. Thus the program modules which perform those functions may be considered to be a "non-resident executive subsystem".

There are functions to be performed in the Technology Complex (Figure 18) which are executive in nature and could be considered to be a "resident executive subsystem". Some of these functions are apparent in Figure 19 which illustrates the flow of data from and to external files.

EXTERNAL DATA FLOW

The flow of data between the modules of Figure 18 and the RDF (run data file), the TM (technical module library), the MDB (master data base), and the TDF (temporary data files) are of significance in the design. These are shown in Figure 19 and are described below.

The "preprocess data" module first reads from the RDF the controls describing the technical module to be used.

The technical preprocessing module is then loaded into core which reads input data from the appropriate records of the RDF. Output goes to MDB (optional) and the temporary data file. One such operation may be performed at a time and the operations and module loadings are repeated until this phase is completed.

The "define system" module establishes system degrees of freedom and transformation data. As above, the control information on the RDF specifies which C---D modules are to be loaded from the technical module library (TM). Data are then read from the RDF and in some cases from the temporary data files. These modules may be loaded and executed one at a time. Certain routines which are used for the system definition must reside in main storage while this process proceeds through all the components of the rotorcraft system model being developed. The information developed remains in main storage.

The "form coefficients" module acts in precisely the same manner as above except both C---C and F---C modules are used and different executive routines are required.

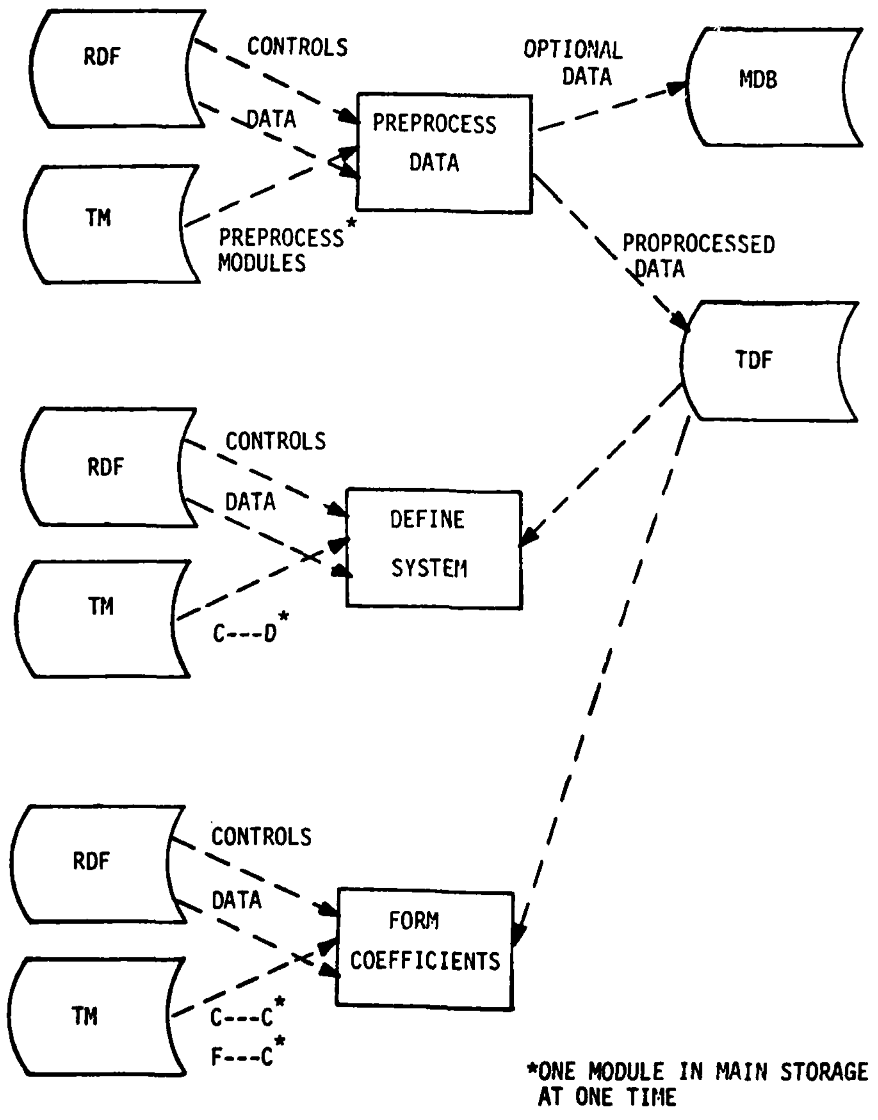
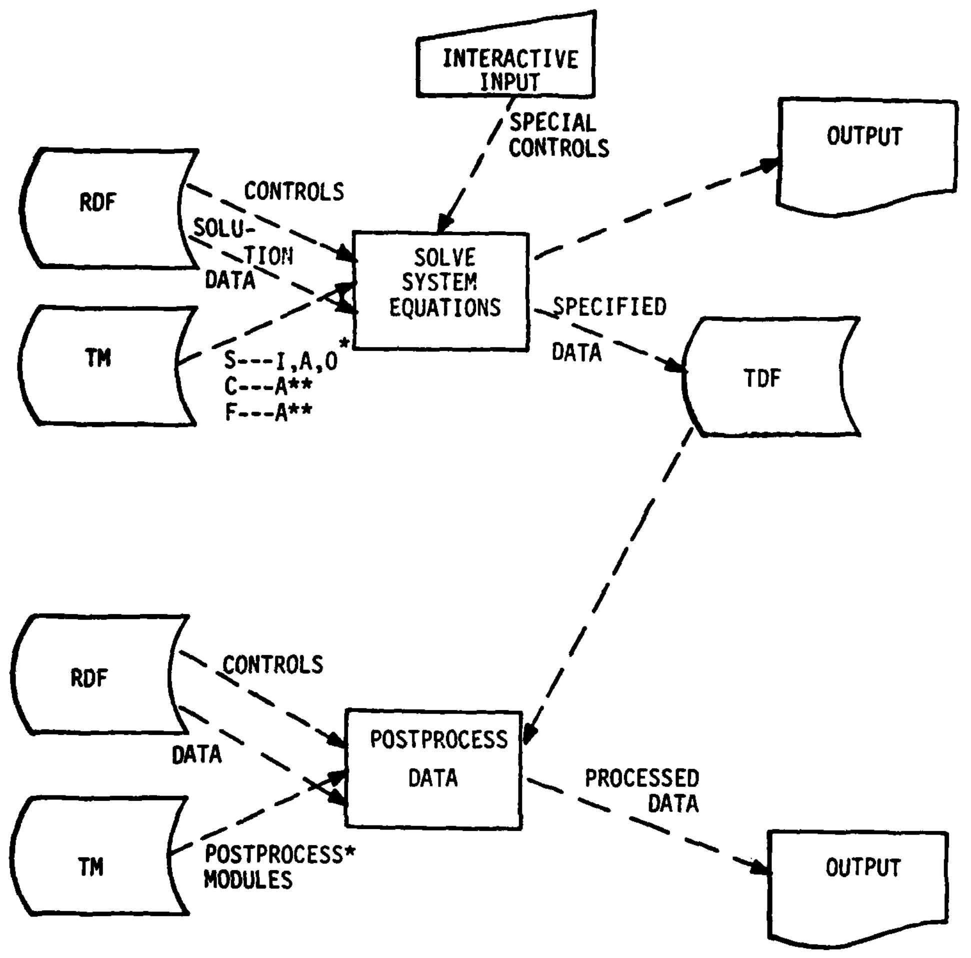


Figure 19. External Data Flow.



*ONE MODULE IN MAIN STORAGE AT ONE TIME
**ALL MODULES IN MAIN STORAGE SIMULTANEOUSLY

Figure 19. (Cont.) External Data Flow.

In the "solve system equations" phase all pertinent C---A and F---A modules must remain in main storage simultaneously since they are used repeatedly. Note that here no component data are read from the RDF except input data related to the solution algorithm. Data may be entered interactively depending on the solution algorithm in use.

The "postprocess data" module is similar to the "preprocess data" module except that it operates on data in the temporary data file and its results are output in report form.

ARCHITECTURAL DESIGN

The architectural design of the coupled analysis (which may be considered to be the "central transform") is shown in Figure 20. The corresponding data dictionary is given below.

There are some characteristics of this design which are noteworthy.

The "interchangeable" modules shown are technology "black boxes". They interface to the Technology Complex through standard (identical) interfaces (argument lists). Therefore, the operation of the Technology Complex is entirely independent of which specific "black boxes" are selected by the user. Also, the development and installation of new modules do not require modification of the Technology Complex.

The sharing of specific data between the executable modules of a technical black box is done through labeled COMMON (or equivalent) as discussed in previous sections (See Figure 10). This data may then be considered to be "local" and does not enter the mainline data flow of the Technology Complex.

As a consequence of the previous two paragraphs, all the data in the mainline paths of the Technology Complex are completely abstract and are not identifiable with particular technical physical characteristics of any of the components or force algorithms. This insures the long range enhancement capabilities of such a system.

The architectural design, as shown, is virtually complete. All the modules shown have been implemented in DYSCO.

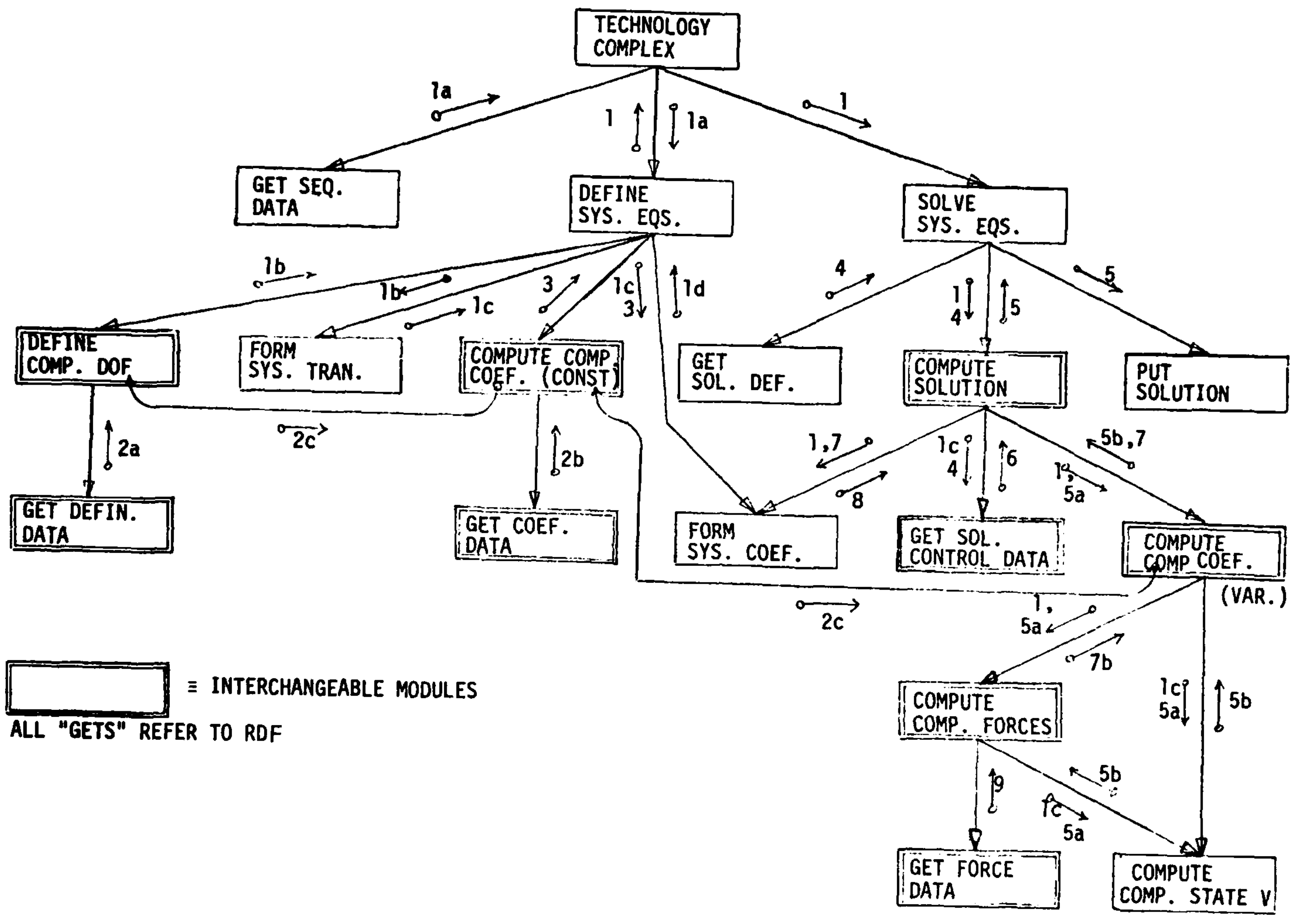


Figure 20. Architecture of Technology Complex Based on DYSCO.

DATA DICTIONARY

The data shown in Figure 20 is defined as follows:

1. System Definition Data

    1.a. Sequence data (identification of components, forces, sources of data)
    1.b. Component DOF names and implicit relationships
    1.c. Transformation data
    1.d. System equation coefficients (constant parts)

2. Component Descriptive Data (specific to each component)

    2.a. Definition data (data required to form 1.b.)
    2.b. Coefficient data (data required to compute 3.)
    2.c. Local component shared data (between "black box" modules)

3. Component Equation Coefficients (constant parts)

4. Solution Definition (identification of solution module)

5. Solution Output (e.g., state vector, time)

    5.a. System state vector
    5.b. Component state vectors

6. Solution Control Data (e.g., initial conditions, flight conditions)

7. Component Equation Coefficients (variable parts)

    7.a. M,C,K matrices
    7.b. F vector

8. System Equation Coefficients (variable parts)

CONCLUSIONS

The DYSCO program has a firm mathematical basis.

The means of specifying the coupling between components and the transformation algorithms are convenient and efficient.

The concept of independent component and force modules lends great flexibility to the applications of the program.

The existing capabilities and the capabilities which may be directly added to the program represent a major portion of existing and advanced rotorcraft dynamic and aerodynamic technology.

An extension of the DYSCO design can result in a system which satisfies all the major goals of a comprehensive helicopter analysis system.

DA
FILM
7–